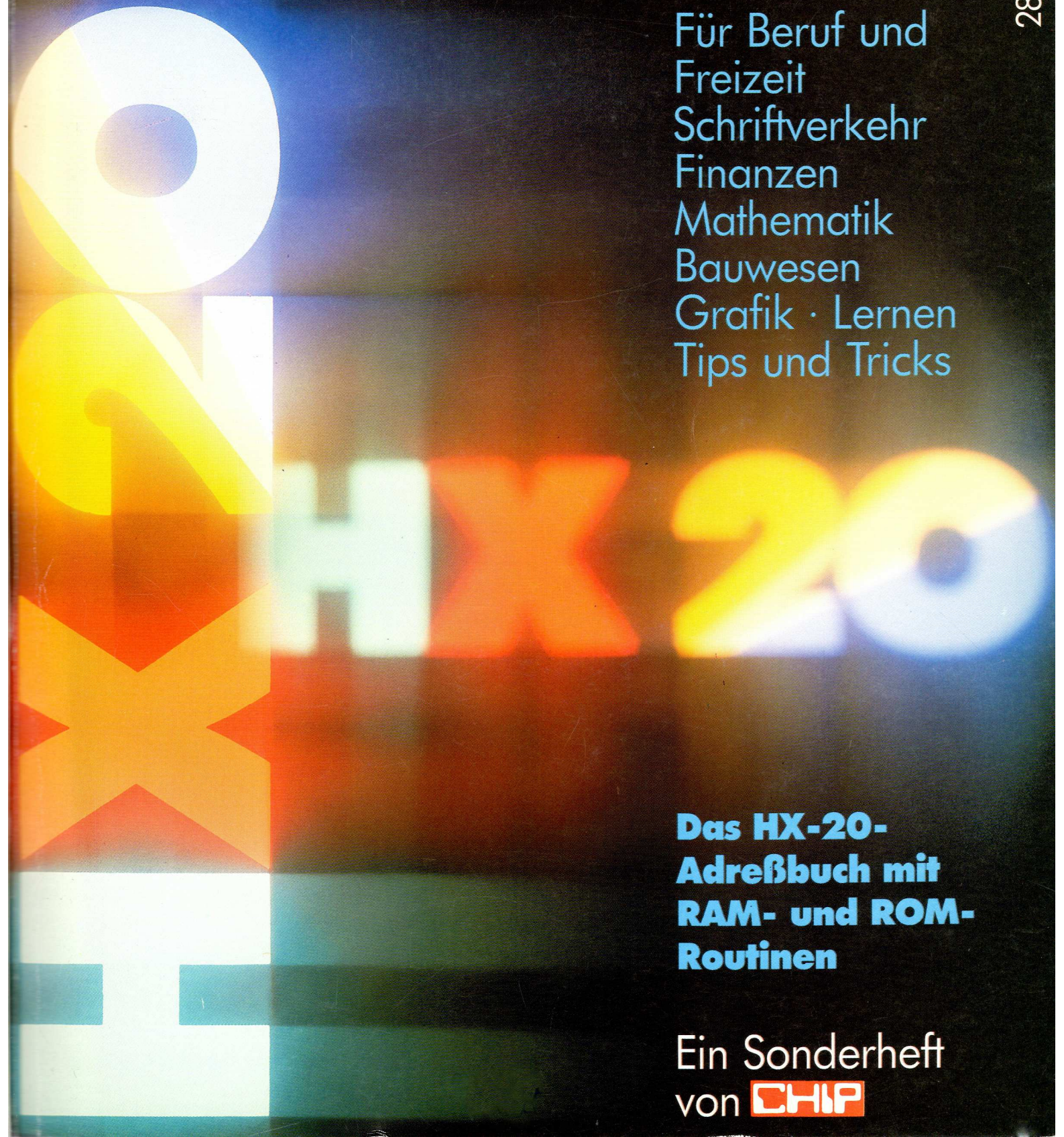
PC-Soft
Anwender-Programme
Epson HX 20

28,– DM 82005/85001

Für Beruf und
Freizeit
Schriftverkehr
Finanzen
Mathematik
Bauwesen
Grafik · Lernen
Tips und Tricks



Das HX-20-Adreßbuch mit RAM- und ROM-Routinen

Ein Sonderheft
von CHIP

Noch mehr Geschmack

Als der HX-20 in der Sparte "Hand-Helds" von Fachjournalisten aus sieben Ländern Ende 1983 zum "Computer des Jahres" erkoren wurde, fragten sich manche Insider: Weshalb ausgerechnet dieser wenig verbreitete Rechner? Wie weitsichtig diese Entscheidung war, zeigt der inzwischen eingetroffene Markterfolg. Unterdessen können sich ebensoviele HX-20-Besitzer in unserem Lande die Hand reichen, wie sie sonst nur bei gängigen Home-Computern zusammen kamen: Der hochwertige tragbare Epson ist dabei, die vom preisgünstigsten Computeranbieter am Jahresanfang 1984 erreichte Stückzahl an Spectrum-Modellen zu überschreiten!
Wer die vielseitigen Anwendungsmöglichkeiten des HX-20 kennt, den wundert das nicht. Aktuelles Beispiel: Wer den HX-20 im Außendienst, auf der Baustelle oder auf Reisen benutzt, kann inzwischen von jeder Telefonzelle aus Daten mit dem Bürocomputer QX-10 austauschen. Voraussetzung ist lediglich, daß der am Arbeitsplatz stehende Bürocomputer an ein Postmodem angeschlossen ist und beide Computer mit einem Programm geladen sind, welches die DFÜ-Routinen enthält (DFÜ = Datenfernübertragung).
Obwohl es ursprünglich unsere Absicht war, die hier exklusiv und erstmalig am deutschen Markt veröffentlichten Programme auch den Besitzern des Schwestermodells PX-8 zugänglich zu machen, haben wir nach intensiven Testläufen davon Abstand genommen. Dazu gibt es zwei Gründe. 1. Obwohl beide Modelle miteinander verwandt sind, hätten sämtliche Programme bis auf eine Ausnahme umgeschrieben werden müssen. 2. Der PX-8 soll und will den HX-20 ja nicht ablösen. Und überhaupt sollen PX-8-Besitzer ja nicht auf eigene Programme verzichten müssen: 1985 ist auch CHIP-Special dabei.

Noch mehr Geschmack am "Gekrönten" wünscht Ihnen mit dieser Programmsammlung

Ihre Redaktion CHIP-Special

„Vor 3 Jahren habe ich mir ein paar Aktien von der Firma gekauft, deren Computer in Chip am besten abgeschnitten hat. Seither begrüßt man mich in meiner Bank voller Hochachtung".

Monat für Monat bei Ihrem Zeitschriftenhändler

Was immer Sie über Mikrocomputer wissen möchten: In Chip, Deutschlands größtem Mikrocomputer-Magazin, finden Sie die Antwort. Dazu in jedem Heft Tests, Tips und Tricks, die Chip-Börse, Kaufberatung und eine Menge mehr. Chip gibt's jeden Monat neu.

CHIP

Chip ist Software für den Kopf.

Inhaltsverzeichnis

Schriftverkehr	6	Briefkopf
	8	Text-Star
	19	Address
Rechnungswesen	22	EDV-Überweisungsvordrucke
Finanzen	28	Ladenkasse
	31	Schuldtilgung
Mathematik	33	Trigonometrische Funktionen
	34	Volumen berechnen
	36	Flächen berechnen
Bauwesen	38	Prismen
Grafik	47	Digitaluhr
	47	Analoguhr
	48	Weihnachtsgruß
Tips und Tricks	50	Zeichengenerierung
	55	HX-Basic
	56	File-Finder
	57	While/Wend-Befehle
Haushaltplan	62	Addition mit Text
Lernen - Wissen - Können	64	Grundrechen-Teacher
Spiel - Sport - Spaß	64	Mein Biorhythmus
	66	Game of Life
	71	Code-Gal
Glücksspiel	73	Lottozahlen ermitteln
	76	PEC-Woman
Party-Nonsen	78	Methusalem
Schach	79	Springer
Musik	80	Loeillet
Programmierhilfen	81	RAM- und ROM-Adressen
Nützliches und Wissenswertes	3	Editorial
	128	So schicke ich Programme ein
	129	Kontakte
	132	Buchtips
	132	Impressum
	133	Abrufkarten
	133	Redaktionsservice

Briefkopf

Programmiersprache: EBASIC. HX-20, Grundversion, Epson-Drucker FX-80, RS-232- Interface 8145, Anschlußkabel# 714. Speicherplatzbedarf: 2544 Bytes für Programm, 2934 Bytes inkl. Variablen.
Dieses Programm ermöglicht den Entwurf eines eigenen Briefkopfes.Kombiniert mit einem Textprogramm oder einem Adressverwaltungsprogramm gibt es dafür eine ganze Reihe von Einsatzmöglichkeiten.
Die Anwenung dieses Programmes setzt keine speziellen Programmierkenntnisse voraus. Zur Umgestaltung eines individuellen Briefkopfes sollten schon Grundkenntnisse zur Programmierung in BASIC vorhanden sein,da Änderungen innerhalb des Programmes erfolgen müssen. Es wurde versucht, die Programmbeschreibung so aufzubauen, daß die wesentlichen Programmänderungen auch ohne BASIC - Kenntnisse vorgenommen werden können.

DAS PROBLEM

Die Grundlage für jede Korrespondenz - ob privat oder geschäftlich - bildet der Kopfbogen. Um hier kostengünstig davon zu kommen, müssen große Stückzahlen in Auftrag gegegben werden. Ändert sich jedoch eine Angabe, wie Adresse, Telefonnummer oder Bankkonto, dann werden alle vorhandenen Bögen zu Altpapier.
Sollte man aber im Besitz eines intelligenten Matrixdruckers sein, dann läßt sich hier eine Menge Geld und Zeit einsparen. Das Druckermodell FX-80 verfügt über einen ladbaren Zeichengenerator, der es zuläßt, alle Zeichen umzudefinieren. Wird das Farbband nicht in einer anderen Farbe gewählt, so ist die Kopfdruckfarbe mit der des Textes identisch.

PROGRAMMBESCHREIBUNG

Das Programm wird nach dem Eingeben mit RUN gestartet. Mit dem OPEN-Befehl in Adresse 20 wird die Schnittstelle RS 232 angesprochen (COM0) und die Datenübertragung spezifiziert (4BN22). Der Datenübertragungsschlüssel lautet:

```
4 = Baud-Rate      : 1200 bps
8 = Wortlänge      : 8-Bit-Wort
N = Parity         : No Parity Check
2 = Stop-Bits      : 2 Stop-Bits
2 = Aktive         : CTS = aktiv
    Kontroll-        DSR = aktiv
    Leitung          RTS = + aktiv
                     CD  = aktiv
```

Dieser Übertragungsschlüssel weicht von der standardmäßig eingestellten Version ab. Damit die übertragenen Daten auch vom Matrixdrucker empfangen werden können, müssen die DIP-Schalter mit dem Datenübertragungsschlüssel aus dem OPEN-Befehl abgestimmt werden. Vergleiche FX-80-Handbuch, Seite 4o bis 46.
Folgende DIP- Schalterstellungen sind herzustellen:

```
Grundplatine FX80     DIP 1:01010101
                      DIP 2:1110
8145-Platine (RS232)  DIP 1:10101111
                      DIP 2:0101
```

1 = ON : 0 = OFF

In Adresse 30 wird die 80- Zeichenbegrenzung für eine Druckerzeile aufgehoben. Adresse 40 bis 260 besorgt den Ausdruck der Schrift. Adresse 270 bis 290 transportiert das bedruckte Blatt wieder an den Anfang zum Ausdruck der Sonderzeichen.
In Adresse 310 wird der vorhandene Zeichensatz des FX-80 in den RAM-Bereich des Druckers kopiert. Erst dann kann mit dem Umdefinieren der Zeichen begonnen werden. In Adresse 330 wird festgelegt, welche Zeichen umdefiniert werden sollen: die ASCII-Zeichen 193 bis 196. Also vier Zeichen (ab ASCII - Tabelle im FX- 80- Bedienungshandbuch, Seite 186).
Die neue Definition für das ASCII - Zeichen 193 = A wird durch die Adressen 340 bis 360 festgelegt, in Adresse 370 bis 390 für das ASCII-Zeichen 194 = B , in Adresse 400 - 420 für C und in Adresse 430 bis 450 für D. In Adresse 46o bis 610 wird der ladbare Generator angewählt und mit den Zeichen A bis D lassen sich die neu definierten Zeichen aufrufen und ausdrucken. Ist der Ausdruck beendet, wird

wieder der normale ROM- Zeichengenerator angewählt. Die Schnittstelle RS 232 wird geschlossen.

DER EIGENE BRIEFKOPF

Zur Textänderung sind folgende Print # 1 - Befehle zu korrigieren:
Adresse 50: die 1. Zeile im gedehnten Fettdruck für den Vor- und Zunamen.
Adresse 80: den Text im Indexmodus für das Sichtfenster.
Adresse 110: die 2. Zeile im Druck für die Wohnstraße.
Adresse 130: die 3. Zeile im Fettdruckfür den Wohnort.
Adresse 180: die 4. Zeile im Fettdruck für die Telefonnummer.
Hierzu werden die Adressen einfach mit dem LIST- Befehl einzeln aufgerufen. Mit dem Cursor rückt man bis zum ersten zu ändernden Buchstaben vor und überschreibt einfach. Eventuell erforderliche Leerräume können später noch dazwischen geschoben werden. Ist eine Adresse fertig, muß mit RETURN abgeschlossen werden .
Zur Graphikänderung sind die Befehle Print # 1 der Adressen 490 bis 61o zu korrigieren. Zur Verfügung stehen bereits vier definierte Sonderzeichen: A, B, C und D. Durch entsprechende Änderung von Lage und Typ der einzelnen Sonderzeichen läßt sich eine Menge neu gestalten. Wenn dies nicht ausreicht, kann man entweder noch zusätzliche Sonderzeichen definieren oder die vorhandenen ändern. Um eine einfache Änderung zu ermöglichen, wurden in den Adressen 360, 390, 420 und 550 die Variablen X und Y verwendet. Man braucht also nur je einen X- und/oder einen Y-Wert in Adresse 360 bzw. 390, 420, 550 zu verändern, um neue Sonderzeichen zu bekommen (siehe hierzu Bitmustermodus im FX-80 Handbuch auf Seite 91/133). Für X oder Y dürfen also keine beliebigen Zahlenwerte eingesetzt werden, sondern nur Werte, die durch die Addition der folgenden Zahlen gebildet werden können von 0 bis 225 : 1, 2, 4, 8, 16, 32, 64, 128. Außerdem müssen einzelne Punkte, wenn sie nebeneinanderliegen, durch den Wert 0 getrennt werden.

Arno W.Wagner

```
10 'BRIEF - Briefkopf
20 'für Chip-Special
30 'von Arno W. Wagner
40 OPEN"O",#1,"COM0:(48N22)"
50 WIDTH"COM0:",255
60 PRINT#1
70 PRINT#1,CHR$(14):CHR$(27);"E";"
                        PETRA  WAGNE
R"
80 PRINT#1
90 PRINT#1
100 PRINT#1,CHR$(27);"S";CHR$(0);CH
R$(27);"M";"          PETRA WAGNER,K
AISER-WILHELM-PL.5,HN";
110 PRINT#1,CHR$(27);"T";
120 PRINT#1,CHR$(27);"P";
130 PRINT#1,CHR$(27);"D";CHR$(48);C
HR$(0);CHR$(9);"Kaiser- Wilhelm- Pl
atz  5"
140 PRINT#1
150 PRINT#1,CHR$(27);"D";CHR$(48);C
HR$(0);CHR$(9);"7100    H E I L B R
 O N N"
160 PRINT#1
170 PRINT#1
180 PRINT#1,CHR$(27);"F";"          "
;A$(1,2);A$(1,3);CHR$(27);"E"
190 PRINT#1,CHR$(27);"F";"          "
;A$(1,4):CHR$(27);"E"
200 PRINT#1,CHR$(27);"D";CHR$(48);C
HR$(0);CHR$(9);"Telefon  :  07131 /
 84187"
210 PRINT#1
220 PRINT#1
230 PRINT#1,CHR$(27);"F";
240 PRINT#1
250 PRINT#1
260 PRINT#1,CHR$(27);CHR$(15);"
      Ihre Zeichen        Ihre Nac
hricht     Mein Zeichen     Datum"
270 PRINT#1,CHR$(18);
280 PRINT#1,"-"
290 FORI=1TO18
300 PRINT#1,CHR$(27);"j";CHR$(36);
310 NEXTI
320 REM ZEICHENSATZ IM RAM KOPIEREN
330 PRINT#1,CHR$(27);":";CHR$(0);CH
R$(0);CHR$(0);
340 REM ZEICHEN UMDEFINIEREN
350 PRINT#1,CHR$(27);"&";CHR$(0);CH
R$(193);CHR$(196);
360 PRINT#1,CHR$(139);
370 X=170:Y=0
380 PRINT#1,CHR$(X);CHR$(0);CHR$(Y)
```

```
;CHR$(0);CHR$(X);CHR$(0);CHR$(Y);CH
R$(0);CHR$(X);CHR$(0);CHR$(Y);
390 PRINT#1,CHR$(139);
400 X=170:Y=85
410 PRINT#1,CHR$(X);CHR$(0);CHR$(Y)
;CHR$(0);CHR$(X);CHR$(0);CHR$(Y);CH
R$(0);CHR$(X);CHR$(0);CHR$(Y);
420 PRINT#1,CHR$(139);
430 X=255:Y=255
440 PRINT#1,CHR$(X);CHR$(0);CHR$(Y)
;CHR$(0);CHR$(X);CHR$(0);CHR$(Y);CH
R$(0);CHR$(X);CHR$(0);CHR$(Y);
450 PRINT#1,CHR$(139);
460 X=255: Y=0
470 PRINT#1,CHR$(X);CHR$(0);CHR$(Y)
;CHR$(0);CHR$(X);CHR$(0);CHR$(Y);CH
R$(0);CHR$(X);CHR$(0);CHR$(Y);
480 REM LADBAREN GENERATOR ANWÄHLEN
490 PRINT#1,CHR$(27);"%";CHR$(1);CH
R$(0);
500 PRINT#1,CHR$(27);"4";CHR$(27);"
3";CHR$(24);CHR$(27);"U";CHR$(1);
510 PRINT#1
520 PRINT#1,"  AAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAA"
530 PRINT#1,"AAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAA"
540 PRINT#1,"AAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAA  "
550 PRINT#1,"AAAAAACCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCAAAAA"
560 FORI=1TO15
570 PRINT#1,"AAAAAA
                    AAAAAAAA
                     AAAAA"
580 NEXTI
590 PRINT#1,"AAAAAACCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCAAAAA"
600 PRINT#1,"  AAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAA";
610 PRINT#1,"
                              AAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA"
620 PRINT#1,"
                              AAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA"
630 PRINT#1,"
                              AAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAA   "
640 PRINT#1,CHR$(27);"5";CHR$(27);"
2";CHR$(27);"U";CHR$(0);
650 REM ROM-ZEICHEN GENERATOR WÄHLE
N
660 PRINT#1,CHR$(27);"%";CHR$(0);CH
R$(0);
670 CLOSE#1
```

Text-Star

Programmiersprache: EBASIC und PASCAL. Gerätekonfiguration: HX20, 32K-Version. FX-80-Matrixdrucker anschließbar. Speicherplatzbedarf: 14450 Bytes.

Briefe schreiben, insbesondere Schemabriefe, Etiketten drucken und fremde Zeichensätze definieren lassen sich mit diesem komfortablen Textverarbeitungsprogramm. "Text-Star" in Verbindung mit dem ebenfalls in dieser Ausgabe veröffentlichten "Adress"-Programm ermöglicht das Erstellen einer umfangreichen Datei.

Das hier abgedruckte Programm ist bildschirmorientiert, weshalb ein Aufbau im DIN-A4-Format gewählt wurde. Um Speicherplatz zu sparen, ist der Bildschirmspeicher gleichzeitig Textspeicher. Das Menü ist von jeder Stelle aus aufrufbar (CHR$(127)). Die CLR-Taste ist aus Sicherheitsgründen gesperrt. Sonderfunktionen sind unkompliziert über CTRL abrufbar. Diese Forderungen bedingen eine zeichenweise Eingabe über INPUT$(1). Um die Befehle PUT% und GET% auf die Bildschirmzeile anwenden zu können, wurde die RAM-File-Startadresse auf die Adresse des Bildschirmanfangs gesetzt und ein RAM-File mit der Satzlänge der Bildschirmbreite definiert.

PROGRAMMAUFBAU

In den Zeilen 970 bis 1070 finden sich: Grundparameter, Definition der Variablen. In der Zeile 1080: Verändern der Systemadressen. Zeile 1090: Maske gespeichert, kann nach belieben geändert werden. Zeilen

1110 bis 1320: Eingabe, hier können Funktionen die mit den CTRL Kommandos zusammenhängen ergänzt oder gelöscht werden. Zeilen 1330 bis 1430: Menü. Jeder Buchstabe bewirkt ein Sprung (Zeile 1420). Die Reihenfolge richtet sich nach dem Alphabet. Wollen Sie eine Funktion löschen, so ist das dazugehörige Sprungziel auf 1430 zu ändern. Zeilen 1440 bis 3380: Textfunktionen. Sie können nach belieben Funktionen löschen, die Kommentare erübrigen eine Beschreibung an dieser Stelle. Zeilen 3390 bis 3490: In diesem Unterprogramm wird eine Zeile interpretiert, die auf den Drucker geschickt werden soll. Wenn Sie einen anderen Drucker verwenden, muß hier geändert werden. Zeilen 3500 bis 3650: Zeile verarbeiten, Worterkennen, Zeilen 3790 bis 3920: Trennen.

EINGABE

Nach dem Eintippen und Starten des Programms befindet man sich im Eingabeteil. Die Eingabemaske ist im Programm voreingestellt (Zeile 1090) mit Rand links ="<", Rand rechts =">", Randausgleich ="!" und TAB Stops ="^". Der linke Rand ist die Position, an der der Cursor nach einer Zeilenschaltung steht. Über den rechten Rand hinaus kann auf keinen Fall geschrieben werden. Randausgleich bestimmt die Position, an der der Beep ertönt und infolge eines Leer- oder "-" Zeichens die Zeile umgebrochen wird.

In der Zeile Null werden die Spalten (1=10, 2=20 usw.), TAB-Stopps, linker Rand, rechter Rand und Randausgleich angezeigt. Diese Zeichen können direkt in der Zeile geändert werden. Gespeichert sind sie nach einem Sprung ins Menü. In der Spalte Null wird die Zeilennummer angezeigt. Diese Zahlen sollen nicht überschrieben werden, weil sie nach einem Befehl im Menü erneut geschrieben werden.

Nur der virtuelle Bildschirm ist Speicher für die eingetippten Buchstaben. Er ist mit 83 Spalten * 72 Zeilen voreingestellt, kann aber nach belieben verändert werden. Alles, was auf dem Bildschirm steht, mit Ausnahme der Zeilen- und Spaltenzahlen, ist gespeicherter Text.

Das Programm arbeitet nach dem gleichen Prinzip, wie der HX-20-Programmeingabeeditor, mit dem Unterschied, daß eine Zeile nicht mit RETURN abgeschlossen werden muß. Alle Cursorsteuertasten und sonstigen Sondertasten können frei genutzt werden. Die HX-20-Editor-CTRL-Sequenzen wurden um einige andere ergänzt, so daß folgende Sonderfunktionen im Eingabeteil zur Verfügung stehen:

CTRL A bewegt den Cursor in die Spalte Null.

CTRL B bewegt den Cursor zum Anfang des nächsten Absatzes.

CTRL E löscht den Rest der Zeile.

CTRL G kehrt die Schreibrichtung um.

CTRL J bewegt den Cursor zum Anfang des nächsten Wortes.

CTRL L (=HOME) bewegt den Cursor zum Ende der Zeile.

CTRL V bewegt den Cursor zum Anfang des vorigen Wortes.

CTRL Z löscht den Rest des virtuellen Bildschirms und setzt den Zeilenzeiger auf die aktuelle Position.

Arbeitet der HX-20 mit einem FX-80-Drucker, gibt es die Möglichkeit, in einem Text verschiedene Schriften zu verwenden. Beim FX-80-Drukker können einige Schriftarten über eine zweistellige Zahl angewählt werden. Wenn diese Zahl in den ersten zwei Spalten einer Zeile steht, wird von dieser Zeile an in der entsprechenden Schrift gedruckt.

Wenn im Drucker-Menü die HX-20-Sonderzeichen auf den Drucker übertragen wurden, können auch Grafikzeichen im Text verwendet und gedruckt werden. Das sind Zeichen, die mit Hilfe der GRAPH- oder CTRL-Taste erzeugt werden.

Soll ein Wort unterstrichen werden, so ist vor und nach dem Wort ein tiefer Strich zu setzen. Dieses Zeichen vor und nach dem Wort wird nicht mit ausgedruckt.

Analog dazu wird mit "↓" (GRAPH.) vor, und "↑" (GRAPH ö) nach dem Wort der Index Druck aktiviert.

"↑" und "↓" ist das Zeichenpaar für Potenzdruck.

"o" (GRAPH z) und "o" (GRAPH u)

aktiviert die Breitschrift.
"o" und "o" aktiviert die Schmalschrift.
Mischungen aus diesen fünf diversen Möglichkeiten in einer Zeile, sind nach belieben möglich.
Tritt ein Zeichen (z.B. ↓) ohne seinen Nachfolger (↑) in einer Zeile auf, wird in dem Text dieses Zeichen wie jedes andere gedruckt. Dadurch ist es möglich, in Texten mit diesen Grafikzeichen zu arbeiten. Wenn Sie Formeln mit Indizies und Potenzen auf Bruchstrichen schreiben wollen, ist es am einfachsten, wenn Sie diese erst einmal ohne Steuerzeichen schreiben und diese später einzufügen. Dadurch gibt es keine Probleme mit Abständen.
Neben dieser komfortablen Eingabe gibt es noch eine große Anzahl von Textfunktionen. Diese werden mit der Taste GRAPH "∧" aufgerufen. In der Zeile Null wird ein Menü angezeigt. Es besteht aus folgenden Zeichen (ABCDEFGHIJKLMNOPQRSTUVXZ<> ? ∧). Jedes steht für die Abkürzung einer Funktion. Der Dialog zu den einzelnen Funktionen wird ebenfalls in der Zeile Null durchgeführt.

Eingabeeinheit	Ausgabeeinheit	
BRCD:	--------	= Lesestift
CASO:	CASO:Name.ART	= eingebaute Mikrokassette
CAS1:	CAS1:Name.ART	= externer Kassettenrecorder
COMO:	COMO:	= RCS-232c-Schnittstelle
KYBD:	--------	= Tastatur
--------	LPTO:	= Minidrucker
PACO:	--------	= ROM Cartrige
--------	CSRN:	= Bildschirm
A:Name.ART	A:Name.ART	= Floppy Disk A
B:Name.ART	B:Name.ART	= Floppy Disk B
C:Name.ART	C:Name.ART	= Floppy Disk C
D:Name.ART	D:Name.ART	= Floppy Disk D

A bsatz ordnen (+ Formatieren 1992 BYTES)
Von der aktuellen Cursorposition (=Cursorposition bei Menüaufruf) wird der Text nach den Maskenspezifikationen neu geordnet. Weil nur mit dem Bildschirmspeicher gearbeitet wird, rutscht der Text eine Zeile auseinander. Wenn ein Wort nicht in den vom Randausgleich bestimmten Raum paßt, muß getrennt werden. Dazu wird das Wort in ganzer Länge ausgedruckt, an der Stelle, wo spätestens getrennt werden muß, wird ein "." gedruckt. Gehen Sie nun mit der < --- Taste zum Ende der letzten Silbe und tippen ein "-" ein. Das Wort wird automatisch getrennt. Falls das ganze Wort in die nächste Zeile muß, geben Sie an Stelle des "-" ein " " (=Space) ein.
B locksatz (380 BYTES)
Der Text wird in Blocksatz ge-

Soll mit der 16K-Version des HX-20 gearbeitet werden, so muß man sich aus den angebotenen Funktionen einen Teil aussuchen und über die MERGE-Funktion zum TEXT-STAR-Rumpf dazuladen. Die Anzahl der benötigten BYTES sind bei der Funktionsbeschreibung vermerkt. Es empfielt sich dann die Menüzeile entsprechend abzuändern.
Zur Berechnung des Speicherplatzes folgende Angaben: Der virtuelle Bildschirm benötigt X*Y BYTES (bei WIDTH 83,72 5976 BYTES). Der TEXT-STAR-Rumpf benötigt 4140 BYTES für den Programmtext, 800 BYTES für Stringvariablen und 200 BYTES für die numerische Variablen. Die zusätzlichen CTRL-Sequenzen benötigen 834 BYTES. Potenzen und Index 366 BYTES, Schmal- und Breitschrift 348 BYTES, Unterstrich 186 BYTES. Wenn bei Kassettenfunktionen der HEADER angezeigt werden soll, sind dafür 200 BYTES vorzusehen.
DIE TEXTFUNKTIONEN
Im Menü wird bei einigen Funktionen nach dem File-Namen gefragt. Um die Namen für alle Möglichkeiten nicht jedesmal wiederholen zu müssen, sind sie hier aufgeführt:

bracht, d.h. wenn zum Beispiel der Rand rechts 70 ist, und eine Zeile bis 67 geht, werden zwischen den ersten Wörtern 3 " " eingefügt.

C opy (Standard)
In den Text wird von einer Eingabeeinheit Text hinter die aktuellen Cursorposition kopiert. Bei CASO: als Eingabeeinheit wird noch nach "wind" gefragt. Geben Sie hier bitte die Bandnummer ein, an der das File beginnt. Die Kassette spult entsprechend.

D ouble (387 BYTES)
Von der aktuellen Cursorposition an kann ein Zeichen oder Wort im Text durch ein anderes x-mal ersetzt werden.

E tiketten (und Gruppen anschreiben 348 BYTES
Es werden auf Adressaufklebern die Adressen gedruckt, die mit dem "ADRESS"-Programm gespeichert wurden, und in denen das gewünschte Kriterium vorkommt. Das Programm ist auf zweibahnige Etiketten 100 35.2 eingestellt (HERMA 8167).

F ormatieren (und Absatz ordnen 1992 BYTES)
Der gesammte Text wird neu formatiert, analog zum Absatz "Ordnen".

G ruppe anschreiben (und Etiketten 1348 BYTES)
Ähnlich wie bei Etiketten muß der Name des Adressfiles und das Kriterium eingegeben werden. Dann werden die Adressen mit dem Text auf dem Bildschirm auf den Drucker gebracht. Der Text auf dem Bildschirm beginnt nach "Sehr geehrter Herr XY,".

H urtig (Standard)
Wenn Sie schneller schreiben, als das Bildschirmscrolling nachkommt, gehen Sie in den H-Mode, die Schreibposition befindet sich nun in der ersten Bildschirmzeile, was die Übersicht allerdings mindert. Im H-Mode wird bei Copy der Text vom linken Rand an geschrieben. Sie können damit neben einen Textblock Text kopieren. Erneutes H im Menü macht den H-Modus rückgängig.

I nsert (Standard)
Mit Insert werden Zeilen in den Text eingefügt, von der aktuelle Cursorposition an die gewünschte Anzahl. Falls die aktuelle Cursorposition "Rand links" ist, wird von der aktuellen Cursorposition an in die nächste Zeile geschoben.

J ump End (Standard)
Der Cursor springt zur letzten Zeile des Textes.

K ürzel listen (505 BYTES)
Es werden die Abkürzungen aus den beiden Menüs auf eine Ausgabeeinheit geschrieben.

L öschen (Standard)
Von der aktuellen Cursorposition an wird die gewünschte Anzahl Zeilen gelöscht. Geben Sie nur RETURN ein, wird eine Zeile gelöscht.

M athematik (362 BYTES)
In der Menüzeile arbeitet das Programm als Vier-Funktionsrechner mit +-/ als Rechensymbol. Zurück mit GRAPH ∧ + RETURN.

N eues Druckformat (1147 BYTES)
Ein neues Menü erscheint, der angeschlossene Drucker kann programmiert werden. Das Programm ist auf den Epson-Matrixdrucker FX-80 eingestellt.

P F Tasten belegen (+ Quitt 503 BYTES)
Die Funktionstasten können mit Text beglegt werden.

Q uittieren (+ PF-Tasten belegen 505 BYTES)
Mit # wird TEXT-STAR beendet, zudem werden die Funktionstasten auf den definierten Standard zurückgesetzt; es wird in das HX-20-Hauptmenü zurückgesprungen.

R aster Druck (240 BYTES)
Der Text von Zeile 1 bis 4 und Spalte 3 bis 23 wird, mit einem gewünschten Zeichen (zum Beispiel oder X), groß auf den Drucker gebracht.

S chreiben (Standard)
Der Text wird auf eine Ausgabeeinheit geschrieben. Wird auf CASO: geschrieben, muß noch die Bandnummer eingegeben werden, zu der die Mikro-Kassette gespult werden soll. Lassen Sie auf SCRN: schreiben, wird der Text von der aktuellen Cursorposition an geschrieben. Der Text wird Zeile um Zeile geschrieben. Tippen Sie nur RETURN, wird vom Anfang bis zum Ende des Textes geschrieben. Geben Sie "M" ein, so wird die Maske gespeichert, bei "copy" wird automatisch eingelesen. Geben Sie "F" ein, so wird als letztes Zeichen "FF" gesendet

(FF=Formularvorschub).
T ext drucken (214 BYTES)
Geben Sie bei "wind" die Bandnummer an, von der an alle folgenden Textfiles gedruckt werden sollen. Nach jedem File wird FF gesendet.
U hr (Standard)
In der Menüzeile wird die Uhrzeit angezeigt.
V ersand Brief (358 BYTES)
Diese Funktion läuft ähnlich wie "Etiketten drucken", nur das in diesem Fall die Adressen mit dem Text auf dem Bildschirm gemischt werden. Lassen Sie den Brief-Text in der ersten Zeile nach "Sehr geehrter Herr XY" beginnen.
Z oom (504 BYTES)
Mit der HX-20-Grafik wird der Text als Ansammlung von Punkten auf dem Display angezeigt. Dadurch kann man die Textstruktur erkennen.
<(>) justieren (280 BYTES)
Der Text wird um X Stellen nach rechts (>) oder links (<) gerückt.
? suchen (Standard)
Von der aktuellen Cursorposition an wird ein String im Text gesucht. Wollen Sie, falls der erste String gefunden wurde, den gleichen noch einmal suchen lassen, dann brauchen Sie nur RETURN eintippen.
G ∧ zurück (Standard)
Mit GRAPH ∧ kommen Sie aus dem Menü zurück in den Text an die aktuelle Cursorposition.
NEUES DRUCKFORMAT (1147 BYTES)
OO Schrift Nummer
Geben Sie die gewünschte Schrift-Nummer mit zwei Ziffern ein (zum Beispiel 04).
A ltern.Zs
Es wird von nun an kursiv gedruckt.
E lite
Es wird von nun an in Elite-Schrift gedruckt.
F orm Feed
Es erfolgt ein Papiertransport auf das nächste Format.
G rafik
Alle HX-20-Grafikzeichen werden auf den Drucker gebracht.
I nkremental
Im Inkremental-Mode zeigt der Drukker die ausgegebenen Zeilen sofort an. Wenn Sie bestimmte Druckbilder austesten wollen, entfällt das lästige Hin- und Herkurbeln von Hand.
L eise
Der Drucker arbeiter leise und langsam.
M ini
Es wird ab jetzt im Index-Mode gedruckt, das heißt jeder Buchstabe ist nur noch maximal vier Nadeln hoch. LF 5 paßt gut zu dieser Schrift.
N ormieren
Diese Funktion setzt "Altern.Zs", "Elite", "Inkremental" und "Mini" zurück.
P roportional ZS
Alle Zeichen werden in Proportionalschrift gedruckt.
S eitenvorschub ändern
Der Seitenvorschub in Zeilen pro Seite kann geändert werden.
U nidirektional
Der Druckkopf druckt nur noch von links nach rechts, dadurch wird minimaler Versatz der Zeichen ausgeglichen. Besonders bei grafischen Darstellungen zu empfehlen.
V erändern LF
Der Zeilenvorschub (Line feed) kann geändert werden. Geben Sie bitte ein, um wieviel Punkte transportiert werden soll (Normal 12, dicht 8, weit 14, Mini 5 oder 4).
ZS USA
Zeichensatz USA wird nun gedruckt.
G∧ zurück
Mit GRAPH ∧ geht es zurück in das Hauptmenü.
WEITERE MÖGLICHKEITEN
Als Drucker wird eine Ausgabeeinheit nur erkannt, wenn diese "COM0:" genannt wurde. "COM0:xy" wird nicht als Drucker erkannt, somit wird das File so über COM0: gesendet, wie es auf dem virtuellen Bildschirm steht.
Header wird der Informationskopf vor einem File auf der Kassette genannt. Wenn im Programm ein File von der Kassette gelesen wird, druckt der Mini-Drucker "Name, Datum und Zeit" dieses Files aus.
PF-Tasten: Brechen Sie das Programm über Q # ab, werden die PF-Tasten neu belegt. Natürlich haben Sie auf Ihrem HX-20 eine andere Standardbelegung der Funktionstasten. Ändern Sie bitte aus diesem Grund die Zeilen 2020 und 2030 ab.
FEHLERMELDUNGEN: Eine spezielle

Fehlerbehandlungsroutine würde für dieses Programm viel Platz beanspruchen und nicht viel bringen. Aus diesem Grund wurde eine globale Fehlerroutine verwendet, die - falls ein Fehler auftritt - die Fehler-Nummer und -Zeile über LPT0: ausgibt. Fehler, die auftreten können, sind zum Beispiel "falscher Gerätename" oder "OM Error" usw. Die Textbearbeitung kann nach Auftreten eines solchen Fehlers ungestört weitergehen. Das Programm kann auch über die BREAK-Taste abgebrochen werden. Wenn Sie erneut starten wollen, ohne den Bildschirminhalt zu verlieren, starten Sie das Programm mit "goto 1000".
ÄNDERUNGEN: Der virtuelle SCRN soll verändert werden. In Zeile 980 sind dazu die Parameter von WIDTH entsprechend abzuändern. Diese Parameter sind auch auf dem XM und YM in Zeile 1060 zu übertragen.

Alexander Broziat

Taste GRAPH +						
q r	w ┬	e ┐	r |	t ─	z ●	u ○
i ♠	o ♥	p ♦	ü ♣	a ├	s ┼	d ┤
f ■	g ▬	h ▮	j ▩	k ♪	l ±	ö ↑
ä ×	y └	x ┴	c ┘	v ▲	b ↗	n ↘
m ▼	, ←	. ↓	- ÷			
selbst definierte Zeichen :						
1 α	2 β	3 γ	4 Δ	5 ε	6 η	7 ϰ
8 λ	9 μ	0 ∂	+ π	# ϱ		
Taste CTRL +						
1 '	2 '	3 °	4 ,	5 :	6 :	7 ◆
8 [	9]	0 ƒ	+ ϱ	# ϒ	, Σ	. ϕ
- ϒ	: {	/ ω	= ↓	? ϑ		

```
950 'TextStar
960 'für Chip-Special
965 'von Alexander Broziat
970 CLEAR 800,0:WIDTH"COM0:",255
980 WIDTH  90,72,4 'XM,YM
990 POKE &H7E,&H80:POKE &H169,4:'POKE 125,4:POKE (PEEK(&H4B5)*256+PEEK(&H4B6)+10
),&H50
1000 ON ERROR GOTO 4110
1010 DEFSTR F,D                          'D,DNEU,FILE,DMY,DM,DSW,DT,DATUM
1020 DEFINT B,H,J,N,P,R,V,W,X,Y,Z   'BIS,HTG,J,J1,J2,JS,JZ,NR,PR,RAAUS,RALI,RARE,
VON,WOANF,WOEND,                                  X,XY00,XM,YM,ZANZ,ZEIZ
1030 DEFSNG I,A,L,C
1040 DEFDBL T                            'TELE
1050 DEFFNDR(X)=CHR$(27)+"!"+CHR$(X)
1060 XM=90 :YM=72:XY00=PEEK(&H270)*256+PEEK(&H271):SCROLL ,,12,4:ZAKT=1:ZANZ=1:
                    DATUM="MoDiMiDoFrSaSo"
1070 DEFFIL XM,0
1080 POKE &H5A2,VAL("&H"+LEFT$(HEX$(XY00),2)):POKE &H5A3,VAL("&H"+RIGHT$(HEX$(XY
00),2))
1090 DT="00  .<   1    .^   2    .^   3    .^   4    .^   5    .^   6   !.^  7>
  .^   8    .^   9    .^   "
1100 '
1110 'Eingabe
1120   RALI=INSTR(DT,"<"):RARE=INSTR(DT,">")-1:RAAUS=RARE-INSTR(DT,"!"):SPAKT=RA
LI:LOCATE 0,0:          PUT% 0,DT
1130   FOR I=0 TO YM ö 10 STEP 0.5:POKE XY00+I*XM*10,INT(48+I):POKE XY00+I*XM*10
+1,(I-INT(I))*          10+48:NEXT I
1140   IF ZAKT>3 AND ZAKT<YM-3 THEN LOCATES 0,ZAKT-3
1150   LOCATE SPAKT,ZAKT
1160     IF CSRLIN=YM-1 THEN PRINT CHR$(30);CHR$(30);:SOUND 28,3
1170     IF CSRLIN>ZANZ THEN ZANZ=CSRLIN
1180     PRINT CHR$(22);:D=INPUT$(1)
1190     IF ASC(D)>31 THEN GOTO 1290
1200     IF D=CHR$(13) THEN PRINT:PRINT STRING$(RALI,28);:IF HTG THEN PRINT STRI
NG$(3,31);                STRING$(3,30);:GOTO 1160 ELSE GOTO 1160
1210     IF D=CHR$(2) THEN GET% CSRLIN+1,DMY:IF RIGHT$(DMY,XM-2)=STRING$(XM-2,32
)THEN PRINT:              PRINT STRING$(RALI,28);:GOTO 1160 ELSE PRINT:GOTO 1200
1220     IF D=CHR$(18) THEN PRINT CHR$(22);:D=INPUT$(1):IF ASC(D)>31 THEN PRINT
```

```
CHR$(18):D;:D=              CHR$(18):GOTO 1220 ELSE GOTO 1190
1230      IF D=CHR$(22) THEN GET% CSRLIN,DM:GOSUB 3740'Wort Anf
1240      IF D=CHR$(7) THEN PRINT CHR$(22);:D=INPUT$(1):IF ASC(D)>31 THEN PRINT D
;:CHR$(29);CHR$              (29);:D=CHR$(7):GOTO 1240 ELSE GOTO 1190
1250      IF D=CHR$(12) THEN GET% CSRLIN,DM:X=INSTR(POS(0)+2,DM,"      "):IF X>0 T
HEN Y=CSRLIN:               LOCATE X-1,Y:IF Y>1 AND X<XM-10 AND X>10 THEN LOCATES X
-10,Y-2:LOCATE X-1,Y:GOTO 1160              ELSE GOTO 1160 ELSE GOTO 1160
1270      IF D=CHR$(10) THEN PRINT CHR$(28);:GET% CSRLIN,D:X=INSTR(POS(0)+2,D," "
):IF X>0 THEN               Y=CSRLIN:LOCATE X,Y:IF Y>1 AND X<XM-10 AND X>10 THEN LO
CATES X-10,Y-2:LOCATE X,Y:GOTO              1160  ELSE GOTO 1160
1280      IF D=CHR$(26) THEN PRINT D;:ZANZ=CSRLIN
1290      IF D=CHR$(127)THEN ZAKT=CSRLIN:SPAKT=POS(0):GET% 0,DT:GOTO 1350'Menue
1300      IF POS(0)<RARE-RAAUS THEN 1330 ELSE IF (D=" "OR D="-") AND CSRLIN>0 THE
N PRINT D::                 D=CHR$(13):GOTO 1190
1310      IF POS(0)=RARE-RAAUS THEN SOUND 20,2
1320      IF POS(0)>=RARE AND CSRLIN>0 THEN SOUND 28,3:PRINT CHR$(29);" ";CHR$(29
);:GOTO 1160
1330      PRINT D;
1340    GOTO 1160
1350 'Menue
1360    FOR I=0 TO YM ö 10 STEP 0.5:POKE XY00+I*XM*10,32:POKE XY00+I*XM*10+1,32:N
EXT I
1370    LOCATE 0,0:IF ZANZ>YM THEN ZANZ=YM
1380    SCROLL 9,0:PRINT CHR$(5);"ABCDEFGHIJKLMNPQRSTVWZ?<>^";:LOCATE 0,0
1390    D=INPUT $(1):LOCATE 0,0:PRINT CHR$(5);:X=0
1400    IF D=CHR$(127) THEN GOTO 1110
1410    IF ASC(D)>96 AND ASC(D)<123 THEN X=ASC(D)-91
1420    IF ASC(D)>59 AND ASC(D)<91  THEN X=ASC(D)-59
1430    IF X>0 AND X<32 THEN ON X GOTO 1730,1440,1750,3060,1440,2570,2380,1970,31
20,2920,1710,               2770,1450,1480,1460,2090,1530,2180,3190,1440,2050,2130,24
90,1810,2680,1470,2290,1440,1440          ,1440,1580
1440 SOUND 28,1:SOUND 25,1:SOUND 28,1:GOTO 1350
1450 IF HTG THEN HTG=0 ELSE HTG=1:GOTO 1110
1460 SPAKT=RALI:ZAKT=ZANZ:GOTO 1110
1470 LOCATE 0,0:SCROLL 9,0:PRINT TIME$;"" ;MID$(DATUM,DAY*2-1
,2);"" ;MID$(DATE$,4,3);LEFT$(DATE$          ,3)
;RIGHT$(DATE$,2):D=INKEY$:IF D="" THEN GOTO 1470 ELSE GOTO 1110
1480 'Insert
1490    LOCATE 0,0:PRINT CHR$(5);"wiev. Zeilen";:INPUT" ",D:IF D="" THEN X= 1 ELS
E X=VAL(D)
1500    IF SPAKT > RALI THEN LOCATE SPAKT,ZAKT:FOR I=SPAKT-RALI+1 TO XM:PRINT CHR
$(18);:NEXT I:          GOTO 1520
1510    FOR J=1 TO X:LOCATE XM-1,ZAKT-1:PRINT CHR$(18):NEXT J:ZAKT=CSRLIN:ZANZ=ZA
NZ+X
1520 GOTO 1110
1530 'Löschen
1540    LOCATE 0,0:PRINT CHR$(5);"wiev.Zeilen";:INPUT" ",D:IF D=""THEN X=1 ELSE X
=VAL(D)
1550    FOR J=ZAKT TO ZANZ :IF J+X< YM THEN GET% J+X,D:PUT% J,D
1560    NEXT J:ZANZ=ZANZ-X:LOCATE 0,ZANZ:PRINT CHR$(26):
1570 GOTO 1110
1580 'Zoom Text
1590    I=XM/31
1600    FOR X=1 TO ZANZ
1610      GET% X,D:Y=2
1620      Y=INSTR(Y,D,"  "):IF Y>0 THEN IF MID$(D,Y,XM-Y)=STRING$(XM-Y," ")THEN Y
=Y/I:X1=119-X:              J=1 ELSE Y=Y+3:GOTO 1620 ELSE Y=XM
1630      IF Y>2 THEN LINE (X1,1)-(X1,Y),PSET
1640      LINE(X1,Y)-(X1,32),PRESET
1650      J=INSTR(J,D,STRING$(I,32)):IF J>0 AND J<Y*I THEN PRESET(X1,J/I):J=J+1:G
OTO 1650
1660    NEXT X
1670    FOR X=0 TO 119-ZANZ:LINE(X,0)-(X,32),PRESET:NEXT X
1680    LINE(119,0)-(119-YM,0),PSET :LINE(119,XM/I)-(119-YM,XM/I),PSET:LINE(119,0
```

```
)-(119,XM/I),            PSET:LINE(118-YM,0)-(118-YM,XM/I),PSET
1690   D=INKEY$:IF D="" THEN GOTO 1690 ELSE GCLS
1700 GOTO 1110
1710 'Formatieren
1720   PR=1:ZAKT=1:GOTO 2590 'Absatz ordnen
1730 '< & >
1740   DMY=CHR$(8):GOTO 1760
1750   DMY=CHR$(18)
1760   PRINT CHR$(5);:INPUT"von ",D:VON=VAL(D):IF VON=0 THEN VON=1
1770   PRINT CHR$(30);CHR$(5);:INPUT"bis ",DM:BIS=VAL(DM):IF BIS=0 THEN BIS=ZANZ
1780   PRINT CHR$(30);CHR$(5);:INPUT"wieviel Spalten ",DM:NR=VAL(DM)
1790   FOR J=VON TO BIS:LOCATE 2,J:PRINT STRING$(NR,DMY):NEXT J
1800 GOTO 1110
1810 'Schreiben
1820   GOSUB 3910'File
1830   PRINT CHR$(5);:INPUT"von Zei ",D:VON=VAL(D):IF VON=0 THEN VON=1
1840   PRINT CHR$(30);CHR$(5);:INPUT"bis Zei ",DM:BIS=VAL(DM):IF BIS=0 THEN BIS=
ZANZ
1850   OPEN"O",#1,FILE
1860   IF FILE="COM0:"THEN PR=1:PRINT#1,CHR$(24); ELSE PR=0
1870   IF FILE="SCRN:"THEN LOCATE 0,ZAKT:SCROLL 9,0
1880   IF D="m" OR D="M" THEN PRINT#1,"Maske"+DT
1890   FOR J=VON TO BIS
1900     GET% J,D:JZ=2
1910     JZ=INSTR(JZ,D,"   "):IF JZ>0 THEN IF MID$(D,JZ,XM-JZ)=STRING$(XM-JZ,32)
 THEN                    D=LEFT$(D,JZ) ELSE JZ=JZ+3:GOTO 1910 ELSE JZ=XM
1920     IF PR THEN GOSUB 3530 ELSE PRINT#1,LEFT$(D,JZ)
1930   NEXT J:IF PR AND(DM="F"OR DM="f") THEN PRINT#1,CHR$(12)
1940   CLOSE #1
1950   IF LEFT$(FILE,5)="CAS0:"THEN LOCATE 0,0:SCROLL 9,0:PRINT CHR$(5);"Bandnr.
";BNR;"-";             TAPCNT;:D=INPUT$(1)
1960 GOTO 1110
1970 'Copieren
1980   GOSUB 3910:OPEN"I",#1,FILE:IF LEFT$(FILE,5)="CAS0:"THEN GOSUB 3840'Header
1990   LINE INPUT#1,D:IF LEFT$(D,5)="Maske" THEN DT=MID$(D,6) ELSE GOTO 2010
2000   LINE INPUT#1,D
2010   IF HTG THEN GET% ZAKT,DMY:D=LEFT$(DMY,RALI)+D:PUT% ZAKT,D ELSE PUT% ZAKT,
D
2020   ZAKT=ZAKT+1:IF NOT EOF(1) THEN GOTO 2000
2030   CLOSE  #1:IF ZAKT>ZANZ THEN ZANZ=ZAKT
2040 GOTO 1110
2050 'PF-belegen
2060   PRINT CHR$(5);
2070   INPUT"PF (1..10) ",NR:PRINT CHR$(30);CHR$(5);:PRINT "PF";NR;".....
...";                  STRING$(15,29);:LINE INPUT D:KEY NR,D
2080 GOTO 1350
2090 'Kürzel list
2100   GOSUB 3910:OPEN"O",#2,FILE:PRINT#2,CHR$(24);:RESTORE
2110   FOR I=1 TO 41:READ D:PRINT#2,D:NEXT I:CLOSE#2
2120 GOTO 1350
2130 'Q uittieren
2140   INPUT"Stop =# ",D:IF D<>"#" THEN GOTO 1350
2150   KEY 1,"WIDTH 80,40":KEY 2,"LIST ":KEY 3,"RUN"+CHR$(13):KEY 4,"SAVE"+CHR$(
34)+"CAS0:";           KEY 5,"LIST"+CHR$(34)+"COM0:"+CHR$(34):KEY 6,"?TAPCNT"+CH
R$(13)+CHR$(30)+CHR$(30)+" "+          CHR$(9)+"=":KEY 7,"WIND "
2160   KEY 9,"TITLE"+CHR$(34):KEY 10,"?TIME$:?DATE$"+CHR$(13):KEY 8,"LOAD"+CHR$(
13)+CHR$(30)+          "TITLE "+CHR$(34)+CHR$(13)
2170 CLEAR 250,0:EXEC &HDFFD
2180 'M athematik
2190   PRINT CHR$(5);:I=0
2200   LOCATE 0,0:PRINT TAB(10);I;CHR$(5):PRINT CHR$(30);:LPRINT I;:INPUT D
2210   IF D=CHR$(127) THEN LPRINT:GOTO 1110
2220   IF D>="*" AND D<="/" THEN LOCATE 0,0:PRINT STRING$(9,32);D:PRINT CHR$(30)
;:INPUT DM:            GOTO 2240
```

```
2230    I=VAL(D):GOTO 2200
2240    LPRINT D;DM:ON ASC(D)-41 GOTO 2250,2260,2200,2270,2200,2280
2250    I=I*VAL(DM):GOTO 2200
2260    I=I+VAL(DM):GOTO 2200
2270    I=I-VAL(DM):GOTO 2200
2280    I=I/VAL(DM):GOTO 2200
2290 'V ersand Brief
2300    LOCATE 0,1:PRINT"24"
2310    LOCATE RALI,6:PRINT "Name";STRING$(4,29);:LINE INPUT"",D
2320    LOCATE RALI,8:PRINT "Straße";STRING$(6,29);:LINE INPUT"",D
2330    LOCATE RALI,10:PRINT "Ort";STRING$(3,29);:LINE INPUT"",D
2340    LOCATE 0,18:PRINT "49 Sehr geehrte";:LINE INPUT"",D:PRINT"24"
2350    LOCATE RARE-22,13:PRINT "Lübeck, den ";
2360    LOCATE RARE-10,13:PRINT MID$(DATE$,4,2):".":MID$(DATE$,1,2);".";"19":MID$
(DATE$,7,2);:           SPAKT=RALI:ZAKT=19
2370 GOTO 1110
2380 'Blocksatz
2390    FOR J=1 TO ZANZ
2400      GET% J,D:J1=2
2410      J1=INSTR(J1,D,"    "):IF MID$(D,J1,XM-J1)<>STRING$(XM-J1,32)THEN J1=J1+3
:GOTO 2410
2420      NR=RARE-J1+1:IF NR>RAAUS+1 THEN GOTO 2470
2430      VON=RALI+1:DMY=" "
2440      FOR BIS=1 TO NR
2450        VON=INSTR(VON,D,DMY):LOCATE VON,J:PRINT CHR$(18);:VON=VON+2:GET% J,D:
IF VON>=J1+BIS              THEN VON=RALI+1:DMY=DMY+" ":BIS=BIS-1
2460      NEXT BIS:DMY=" "
2470    NEXT J
2480 GOTO 1350
2490 'Raster Druck
2500    LOCATE 0,0:PRINT CHR$(5);"Zeichen ";:INPUT D
2510    OPEN"O",#1,"COM0:":WIDTH"COM0:",255:PRINT#1,CHR$(24);FNDR(20);CHR$(27);"U
":LOCATES 3,1
2520    FOR I=0 TO 31:FOR J=0 TO 119
2530      IF POINT(J,I) THEN PRINT#1,D; ELSE PRINT#1," ";
2540    NEXT J:PRINT#1:NEXT I
2550    CLOSE#1
2560 GOTO 1350 'Menue
2570 'Absatz ordnen
2580     PR=0
2590     J=ZAKT+1:JZ=ZAKT:JS=RALI:LOCATE XM-1,JZ-1:PRINT CHR$(18):ZANZ=ZANZ+1:
                    DNEU=STRING$(RALI,32)
2600       GET% J,D:X=2
2610       X=INSTR(X,D,"    "): IF MID$(D,X,XM-X)<>STRING$(XM-X,32) THEN X=X+3:GOT
O 2610
2620       IF X=2 THEN  GOTO 2650 ELSE D=LEFT$(D,X-1)
2630       DMY="":GOSUB 3640'Zeile verarbeiten
2640     J=J+1:GOTO 2600
2650     IF PR THEN IF J<ZANZ THEN LOCATE JS-1,JZ:PRINT CHR$(5):PRINT CHR$(5);:JZ
=JZ+2:J=J+1:            JS=RALI:DNEU=STRING$(RALI,32):IF JZ>=J THEN LOCATE XM-1,
JZ-1:PRINT CHR$(18):J=J+1:GOTO          2600 ELSE GOTO 2600 ELSE PR=0
2660  IF PR=0 THEN LOCATE JS-1,JZ:PRINT CHR$(5):FOR I=JZ+1 TO J:LOCATE 0,I:PRINT
 CHR$(5);:NEXT I
2670 GOTO 1110
2680 'Text schreiben
2690     GOSUB 3910'File
2700     OPEN"I",#2,FILE:GOSUB 3840:OPEN"O",#1,"COM0:":PRINT#1,CHR$(24);
2710      IF NOT EOF(2) THEN LINE INPUT#2,D ELSE GOTO 2740
2720        JZ=LEN(D):GOSUB 3530'COM0: Zeile
2730      GOTO 2710
2740    PRINT #1,CHR$(12)
2750    CLOSE
2760 GOTO 2700
2770 'Gruppe anschreiben
```

```
2780    GOSUB 3910'File
2790    OPEN"I",#2,FILE:OPEN"O",#1,"COM0:":PRINT#1,CHR$(24):
2800    LOCATE 0,0:PRINT CHR$(5);:INPUT"Kriterium ",DMY
2810    IF EOF(2) THEN CLOSE:GOTO 1350
2820      LINE INPUT#2,D:INPUT#2,TELE:J=1
2830      J=INSTR(J,D,DMY):IF J=0 THEN GOTO 2810 ELSE IF J<87 THEN J=J+1:GOTO 283
0
2840      GOSUB 3780'Anschrift
2850      PRINT#1:PRINT#1,TAB(45);"Lübeck den ";MID$(DATE$,4,2);".";MID$(DATE$,1,
2);".19";                  MID$(DATE$,7,2):PRINT#1
2860      PRINT#1,TAB(RALI);"Sehr geehrte";MID$(D,68,20)
2870      FOR J=1 TO ZANZ
2880        GET% J,A:JZ=2
2890        JZ=INSTR(JZ,D,"    "):IF MID$(D,JZ,XM-JZ)=STRING$(XM-JZ,32)THEN D=LEFT
$(D,JZ):GOSUB                3530 ELSE JZ=JZ+3:GOTO 2890
2900      NEXT J:PRINT#1,CHR$(12);FNDR(24):
2910   GOTO 2810
2920 'Etiketten
2930    GOSUB 3910 'File
2940    OPEN"I",#2,FILE:OPEN"O",#1,"COM0:":PRINT#1,CHR$(24);FNDR(24):
2950    LOCATE 0,0:PRINT CHR$(5);:INPUT"Kriterium ",DMY
2960      IF EOF(2) THEN GOTO 3050 ELSE LINE INPUT#2,D:INPUT#2,TELE:J=1
2970      J=INSTR(J,D,DMY):IF J=0 THEN GOTO 2960 ELSE IF J<87 THEN J=J+1:GOTO 297
0
2980      IF EOF(2) THEN GOTO 3050 ELSE LINE INPUT#2,DM:INPUT#2,TELE:J=1
2990      J=INSTR(J,DM,DMY):IF J=0 THEN GOTO 2980 ELSE IF J<87 THEN J=J+1:GOTO 29
90
3000      PRINT#1,TAB(3);MID$(D,1,27);TAB(44);MID$(DM,1,27):PRINT#1
3010      PRINT#1,TAB(3);MID$(D,28,20);TAB(44);MID$(DM,28,20):PRINT#1,FNDR(49)
3020      PRINT#1,TAB(2);MID$(D,48,20);TAB(27);MID$(DM,48,20):PRINT#1,FNDR(24)
3030      PRINT#1:PRINT#1:PRINT#1
3040    GOTO 2960
3050 CLOSE:GOTO 1350
3060 'Suchen
3070    LOCATE 0,0:LINE INPUT"Suchwort: ",DMY:PUT% 0,DT
3080    J=1:IF DMY="" THEN J=ZAKT+1  ELSE DSW=DMY
3090      GET% J,D:JS=INSTR(D,DSW):IF JS > 0 THEN ZAKT=J:SPAKT=JS-1:GOTO 1130
3100    IF J<= ZANZ THEN J=J+1:GOTO 3090
3110 GOTO 1350 'Menue
3120 'D ouble
3130    PRINT CHR$(5);:LINEINPUT"Original ",DMY:PRINT CHR$(30);CHR$(5);:LINE INPU
T"Double ",DSW:          PRINT CHR$(30);CHR$(5);:INPUT"wie oft ",NR
3140    DM="":J=ZAKT:JS=0:J1=1:X=LEN(DMY)-LEN(DSW):IF X<0 THEN DM=CHR$(18):X=ABS(
X) ELSE DM=              CHR$(28)+STRING$(1,8)
3150    GET% J,D
3160      J1=INSTR(J1,D,DMY):IF J1>0 THEN LOCATE J1-1,J:FOR I=1 TO X:PRINT DM;:NE
XT I:PRINT DSW;:          JS=JS+1:J1=J1+LEN(DSW):IF JS<NR THEN GOTO 3150
3170      IF J<ZANZ AND JS<NR THEN J=J+1:J1=1:GOTO 3150
3180 GOTO 1110
3190 'Neue Schrifft
3200    OPEN"O",#1,"COM0:"
3210    PRINT#1,CHR$(24);CHR$(27);
3220    LOCATE 0,0:PRINT CHR$(5);"00AEFGILMNVPSUZ":D=INPUT$(1)
3230    IF D>"/"AND D<":"THEN DM=INPUT$(1):PRINT#1,"!";CHR$(VAL(D)*10+VAL(DM));
3240    IF D="A"OR D="a"THEN PRINT#1,"4";
3250    IF D="E"OR D="e"THEN PRINT#1,"M";
3260    IF D="F"OR D="f"THEN PRINT#1,CHR$(27);CHR$(24);CHR$(12);
3270    IF D="G"OR D="g"THEN PRINT#1,CHR$(27);CHR$(24):GOSUB 3400
3280    IF D="I"OR D="i"THEN PRINT#1,"i1";
3290    IF D="L"OR D="l"THEN PRINT#1,"s";"1";
3300    IF D="M"OR D="m"THEN PRINT#1,"S";"0";
3310    IF D="N"OR D="n"THEN PRINT#1,"5";CHR$(27);"T";CHR$(27);"p0";CHR$(27);"i0"
;CHR$(27);"i0";
3320    IF D="P"OR D="p"THEN PRINT#1,"p";"1";
```

```
3330   IF D="S"OR D="s"THEN LOCATE 0,0:PRINT CHR$(5);:INPUT"Zeilen / Seite ",NR:
PRINT#1,"C";           CHR$(NR);
3340   IF D="V"OR D="v"THEN LOCATE 0,0:PRINT CHR$(5);:INPUT"Punkte je Zeile ",NR
:PRINT#1,"A";          CHR$(NR):
3350   IF D="U"OR D="u"THEN PRINT#1,"U1";
3360   IF D="Z"OR D="z"THEN PRINT#1,"R";CHR$(0);
3370   PRINT#1,CHR$(7);
3380   IF D=CHR$(127) THEN CLOSE#1:GOTO 1350
3390 GOTO 3210
3400 'Grafik
3410   WIDTH"COM0:",255:PRINT#1,CHR$(27);":";CHR$(0);CHR$(0);CHR$(0);
3420   D=CHR$(27)+"%"+CHR$(1)+CHR$(0)+CHR$(27)+"6"+CHR$(27)+"&"+CHR$(0)
3430   FOR K=224 TO 255:GOSUB 3450:NEXT K:FOR K=128 TO 159:GOSUB 3450:NEXT K
3440 RETURN
3450 'Analyse
3460  LOCATE 0,0:PRINT CHR$(K):PRINT#1,D;CHR$(K);CHR$(K);CHR$(139);
3470  FOR X=0 TO 5:ZAHL=0
3480    FOR Y=7 TO 0 STEP-1
3490      IF POINT(X,Y)THEN ZAHL=ZAHL+2^(8-(Y+1))
3500    NEXT Y:PRINT#1,CHR$(ZAHL);:IF X<5 THEN PRINT#1,CHR$(0);
3510  NEXT X
3520 RETURN
3530 'Procedure COM0: Zeile
3540   X=VAL(LEFT$(D,2)):IF X>0 AND X<63 THEN PRINT#1,FNDR(X);
3550   IF INSTR(D,") OR INSTR(D,"") OR INSTR(D,"") OR INSTR(D,"") OR INSTR(D,"_"
) THEN            JZ=JZ-2:D=MID$(D,3,JZ) ELSE PRINT#1,"  ":MID$(D,3):RETURN
3560     JS=2:J1=2
3570     JS=INSTR(JS,D,"):J1=INSTR(J1,D,""):IF JS THEN IF J1>JS THEN D=LEFT$(D,J
S-1)+CHR$(27)+"S0"+MID$(D,JS+1,J1-JS-1)+CHR$(27)+"T"+MID$(D,J1+1):JS=J1:J1=J1+1:
GOTO 3570 ELSE J1=JS:JS=JS+1:GOTO 3570 ELSE JS=2:J1=2
3580     JS=INSTR(JS,D,""):J1=INSTR(J1,D,"):IF JS THEN IF J1>JS THEN D=LEFT$(D,J
S-1)+CHR$(27)+"S1"+MID$(D,JS+1,J1-JS-1)+CHR$(27)+"T"+MID$(D,J1+1):JS=J1:J1=J1+1:
GOTO 3580 ELSE J1=JS:JS=JS+1:GOTO 3580 ELSE JS=2:J1=2
3590     JS=INSTR(JS,D,""):J1=INSTR(J1,D,""):IF JS THEN IF J1>JS THEN D=LEFT$(D,
JS-1)+CHR$(14)+MID$(D,JS+1,J1-JS-1)+CHR$(20)+MID$(D,J1+1):JS=J1:J1=J1+1:GOTO 359
0 ELSE J1=JS:JS=JS+1:GOTO 3590 ELSE JS=2:J1=2
3600     JS=INSTR(JS,D,""):J1=INSTR(J1,D,""):IF JS THEN IF J1>JS THEN D=LEFT$(D,
JS-1)+CHR$(15)+MID$(D,JS+1,J1-JS-1)+CHR$(18)+MID$(D,J1+1):JS=J1:J1=J1+1:GOTO 360
0 ELSE J1=JS:JS=JS+1:GOTO 3600 ELSE JS=2:J1=2
3610     JS=INSTR(JS,D," "):IF JS THEN J1=INSTR(JS+1,D," "):IF J1>JS THEN D=LEFT
$(D,JS-1)+CHR$(27)+"-1"+MID$(D,JS+1,J1-JS-1)+CHR$(27)+"-0"+MID$(D,J1+1):JS=J1:J1
=J1+1:GOTO 3610 ELSE J1=JS:JS=JS+1:GOTO 3610 ELSE JS=2:J1=2
3620     PRINT#1,"  ";D
3630 RETURN
3640 'Procedure Zeile verarbeiten
3650   WOEND=3
3660     GOSUB 3970 'Worterkennen
3670     IF DMY<>"" THEN IF RIGHT$(DMY,1)<>"-"THEN DMY=DMY+" "
3680     IF JS+LEN(DMY)<RARE-RAAUS THEN DNEU=DNEU+DMY:JS=LEN(DNEU):GOTO 3710
3690       DNEU=DNEU+DMY:PUT% JZ,DNEU:IF JS+LEN(DMY)<=RARE THEN JZ=JZ+1:JS=RALI:
DNEU=                        STRING$(RALI,32)  ELSE GOSUB 4040 'Trennen
3700       IF JZ>=J THEN LOCATE WOEND,J:FOR I=WOEND-RALI+1 TO XM:PRINT CHR$(18);
:NEXT I:ZANZ=                ZANZ+1:PRINT CHR$(30);CHR$(30):PRINT:PRINT:PUT% JZ,DN
EU:RETURN
3710     IF RIGHT$(DNEU,1)="-"AND JS<RARE-RAAUS THEN DNEU=LEFT$(DNEU,LEN(DNEU)-1
)
3720   IF WOEND<X THEN GOTO 3660
3730 PUT%JZ,DNEU:RETURN
3740 'Procedure Wort Anf
3750   PRINT CHR$(29);
3760   IF POS(0)>0 THEN IF MID$(DM,POS(0),1)<>" " THEN PRINTCHR$(29);:GOTO 3760
3770 RETURN
3780 'Procedure Anschrift
3790   PRINT#1
```

```
3800   PRINT#1,TAB(RALI);MID$(D, 1,27):PRINT#1 'Name - Firma
3810   PRINT#1,TAB(RALI);MID$(D,28,20):PRINT#1 'Straße
3820   PRINT#1,TAB(RALI):MID$(D,48,20):PRINT#1 'Stadt
3830 RETURN
3840 'Procedure Header
3850   X=VARPTR(D)
3860   POKE X,100:POKE X+1,3:POKE X+2,36
3870   LPRINT MID$(D,9,11);" ";:PRINT CHR$(5);MID$(D,9,11);
3880   D=MID$(D,INSTR(18,D,"      ")+5,12)
3890   LPRINT MID$(D,3,2);".";MID$(D,1,2);".";MID$(D,5,2);TAB(12);MID$(D,7,2);":
";MID$(D,9,2)
3900 RETURN
3910 'Procedure File Name
3920   LOCATE 0,0:PRINT CHR$(5);"File Name ";:LINE INPUT FILE
3930   FOR I=1 TO 4:DM=MID$(FILE,I,1):IF ASC(DM)>91 THEN MID$(FILE,I,1)=CHR$(ASC
(DM)-32)
3940   NEXT I:IF LEFT$(FILE,5)="CAS0:"THEN LOCATE 0,0:PRINT CHR$(5);TAPCNT;:INPU
T"Wind ",BNR:          WIND BNR
3950   IF INSTR(FILE,":")=0 THEN GOTO 3920 ELSE LOCATE 0,0
3960 RETURN
3970 'Procedure Worterkennen
3980   IF MID$(D,WOEND,XM-1-WOEND)=STRING$(XM-1-WOEND,32) THEN DMY="":WOEND=XM:R
ETURN
3990   IF WOEND<XM-1 AND MID$(D,WOEND,1)=" "THEN WOEND=WOEND+1:GOTO 3990
4000   WOANF=WOEND:WOEND=INSTR(WOEND,D," ")
4010   IF WOEND=0 THEN WOEND=LEN(D)+1
4020   DMY=MID$(D,WOANF,WOEND-WOANF)
4030 RETURN
4040 'Procedure Trennen
4050   SOUND 10,1:JS=LEN(DNEU):IF JS<XM-15 AND JZ>2 THEN LOCATES JS-10,JZ-2
4060   LOCATE JS,JZ:PRINT STRING$(JS-RARE+1,29);".";CHR$(29);
4070   DM=INPUT$(1):PRINT DM;:IF DM<>"-" AND DM<>" " THEN PRINT".";CHR$(29);:GOT
O 4070
4080   J1=POS(0):PRINT CHR$(5)
4090    DNEU=STRING$(RALI,32)+RIGHT$(DMY,JS-J1+1):JZ=JZ+1:JS=LEN(DNEU):IF RIGHT$
(DNEU,1)="-"            THEN DNEU=LEFT$(DNEU,JS-1)
4100 RETURN
4110 LOCATE 0,0:SOUND 28,1:SOUND 25,1:SOUND 28,1:PRINTCHR$(5);"Fehler";:CLOSE:LP
RINT"F";ERR;          " in";ERL:SOUND 25,1:SOUND 28,1:SOUND 25,1:PRINTSTRING$(6,8
):RESUME 1110
4120 DATA A bsatz ordenen,B locksatz ,C opy,D ouble ,E tiketten ,F ormatieren,G
ruppe anschreiben ,H urtig,I nsert,J ump End ,K ürzel listen ,L öschen,M athemat
ik ,N eues Druckformat ,P F-Tasten
4130 DATA Q uittieren ,R aster Druck ,S chreiben,T ext drucken ,U hr ,V ersand B
rief ,Z oom *,? suchen ,< justieren,> justieren,G^ zurück." ",00 Schrift Nr,A lt
ern. ZS,E lite,F orm Feed,G rafik,I nkremental
4140 DATA L eise,M ini,N ormieren,P roportional Zs,U nidirektional,V erändern Lf
,Z S USA,G^ zurück
```

Address

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 2585 Bytes. Rund 300 Adressen lassen sich mit diesem Programm in einem File abspeichern. Es eignet sich zum Schreiben von Schemabriefen und zum Ausdrucken von Adress-Etiketten. Nach dem Eintippen startet das Programm mit der Aufforderung, die Kassette einzulegen. Tippen Sie RETURN. Nun geben Sie die Bandnummer ein, ab der die Adressen gespeichert wurden. Falls noch keine Adressen auf dem Band sind, eröffnen Sie ein File mit OPEN "O", #1, "CAS0:ADRESS.DAT".Sie schließen diese Datei mit CLOSE#1. Der Rechner liest die Adressen vom Band und meldet sich mit einer Eingabemaske.

In der ersten Zeile wird die Nummer des zu bearbeitenden Datensatzes angezeigt. Wenn Sie an dieser Stelle die Taste drücken, erscheint der Vorgänger, mit ↓ der Nachfolge . SCRN ↑ springt 10 Datensätze zurück, SCRN ↓ 10 Datensätze vor. Tippen Sie ein S ein, erscheint die Frage "Sprung wohin". Sie können die Nr. des gewünschten Datensatzes eingeben. Tippen Sie die Leertaste, können Adressen eingegeben werden. Die Länge der Eingaben wird durch ■ ←begrenzt. Sind die Eingaben länger, wird alles ignoriert, was hinter dem Balken steht.

NAME

Geben Sie hier die erste Zeile der Anschrift ein. Zum Beispiel: Firma Jessen & Lenz ■; Frau Müller ■ ; usw.

Die Adressen werden später im Programm nach dieser Zeile sortiert. Bedenken Sie dies bei der Eingabe.

STRAßE =

Geben Sie die Straße des Empfängers ein. Zum Beispiel: Wahmstraße 12 ■; usw.

ORT =

An dieser Stelle wird der Ort mit der Postleitzahl eingegeben.

Zum Beispiel: 2400 Lübeck ■ ; usw.

ANREDE =

Mit der Anrede ist der Satz gemeint, der nach " Sehr geehrte " kommt.

Zum Beispiel: Damen und Herren! ■ ; Herr Dr. Müller, ■ ; Frau Müller, ■ ;usw.

KRITERIUM =

Um später eine bestimmte Gruppe anschreiben zu können, ist hier die Möglichkeit, ein Suchkriterium zu definieren.

Zum Beispiel: Lw1■; Bfm■; XXa■;usw.

TELEFON =

Zu guter Letzt kann die Telefonnummer eingegeben erden. Die 0 der Vorwahl erscheint nicht beim Ausdrucken.

Zum Beispiel: 451705030 ■; 401213141 ■; usw.

Wenn Sie die Eingabe der Telefonnummer mit RETURN abgeschlossen haben, ist die Adresse gespeichert, die nächste kann eingegeben werden. Wenn Sie bei einer Eingabe bemerken, daß Sie sich vertippt haben, können Sie unter Benutzung der grauen Tasten korrigieren. Wie beim EDITOR ist der Bildschirm Speicher. Wollen Sie keine Adresen mehr eingeben, tippen sie - sobald die Nummer angezeigt wird- GRAPH ^. Der Rechner sortiert nun die Adressen alphabetisch. Danach kommt die Frage, ob die Adressen gedruckt werden sollen. Wenn gedruckt werden soll, eine 1 eingeben, sonst eine 0.Zuletzt müssen die Adressen noch gespeichert werden. Geben Sie die Bandnummer an. Nach dem Speichern und Drucken wird erneut nach "Drukken" und "Speicher" gefragt, aus Sicherheitsgründen kann somit die Datei nochmals gespeichert werden. Brechen Sie das Programm mit der BREAK-Taste ab.

Alexander Broziat

```
1000 'ADRESS
1010 'für Chip-Special von Alexander Broziat
1020 CLEAR 16000,0:WIDTH 40,7
1030 POKE &H7E,&H80
1040 DEFSTR A,B,D,F,K,N,S 'ANREDE,BUCHST,DMY,FILE,FIRMA,KRIT,STADT,STRASSE
1050 DEFINT I,J,M,X,R,L 'IANZ,MANZ,XY00
1060 DEFDBL T           'TELE
1070 '
1080 XY00=32480 :DMY1=STRING$(4,18):DMY2=STRING$(4,8)
1090 MANZ=200:DIM NAME(MANZ),TELE(MANZ)
1100 '
1110 'Einlesen
1120  PRINT "Kasette einlegen":DMY=INPUT$(1)
1130  INPUT"wind ",J:WIND J:OPEN"I",#2,"CAS0:ADRESS.DAT":I=1
1140  IF EOF(2) THEN GOTO 1170
1150    INPUT#2,NAME(I),TELE(I):I=I+1
1160    GOTO 1140
1170  CLOSE#2:IANZ=I
```

```
1180 'Ändern
1190   IF I>IANZ THEN IANZ=I
1200   IF I<0 THEN I=0
1210   IF I>MANZ THEN I=MANZ
1220   NAME=MID$(NAME(I),1,27):STRASSE=MID$(NAME(I),28,20):STADT=MID$(NAME(I),48
,20):ANREDE=           MID$(NAME(I),68,20):KRIT=MID$(NAME(I),88,3):TELE=TELE(I)
1230   CLS:ZEILE=0
1240   LOCATE 0,0:SCROLL 9,0:PRINT"Name   :";NAME;TAB(34);"
1250   LOCATE 0,1:SCROLL 9,0:PRINT"Straße:";STRASSE;TAB(27);"
1260   LOCATE 0,2:SCROLL 9,0:PRINT"Ort    :";STADT;TAB(27);"
1270   LOCATE 0,3:SCROLL 9,0:PRINT"Anrede:";ANREDE;TAB(27);"
1280   LOCATE 0,4:PRINT"Krit   :";KRIT;TAB(10);"
1290   LOCATE 0,5:SCROLL 9,0:PRINT"Tele   :";TELE;TAB(19);"
1300   LOCATE 0,0:PRINT DMY1;:PRINT USING"### ";I;:SOUND 0,8:DMY=INKEY$:IF DMY="
" THEN PRINT           DMY2;:GOTO 1300 ELSE PRINT DMY2;
1310   IF DMY=CHR$(30) THEN I=I-1 :GOTO 1180
1320   IF DMY=CHR$(31) THEN I=I+1 :GOTO 1180
1330   IF DMY=CHR$(16) THEN I=I-10:GOTO 1180
1340   IF DMY=CHR$(17) THEN I=I+10:GOTO 1180
1350   IF DMY="S"OR DMY="s"THEN CLS:INPUT"Sprung nach ",I:GOTO 1180
1360   IF DMY=CHR$(127)THEN GOTO 1510'Heapsort
1370   LOCATE 0,ZEILE:LOCATE 7,ZEILE
1380   DMY=INPUT$(1):PRINT DMY;CHR$(22);
1390     IF DMY=CHR$(13) THEN ZEILE=ZEILE+1:IF ZEILE<6 THEN GOTO 1370
1400     IF DMY=CHR$(30) AND ZEILE>0 THEN ZEILE=ZEILE-1:GOTO 1370
1410     IF DMY=CHR$(31) AND ZEILE<4 THEN ZEILE=ZEILE+1:GOTO 1370
1420   IF ZEILE<6 THEN GOTO 1380
1430   X0=VARPTR(NAME):XY01=INT(XY00/256)
1440   POKE X0  ,240
1450   POKE X0+1,XY01
1460   POKE X0+2,CINT((XY00/256-XY01)*256)
1470   NAME(I)=MID$(NAME,8,27)+MID$(NAME,48,20)+MID$(NAME,88,20)+MID$(NAME,128,2
0)+                    MID$(NAME,168,3)
1480   TELE(I)=VAL(MID$(NAME,208,12))
1490   I=I+1
1500 GOTO 1180
1510 'Heapsort
1520   IF I>IANZ THEN IANZ=I
1530   L=INT(IANZ/2)+1:R=IANZ
1540   IF R<2 THEN GOTO 1620
1550   IF L>1 THEN L=L-1:J=L:GOTO 1570
1560   SWAP NAME(1),NAME(R):SWAP TELE(1),TELE(R):R=R-1:J=1
1570   IF 2*J>R THEN 1540
1580   I=2*J
1590   IF I<R AND NAME(I)<NAME(I+1)THEN I=I+1
1600   IF NAME(J)<NAME(I) THEN SWAP NAME(J),NAME(I):SWAP TELE(J),TELE(I):J=I:GOT
O 1570
1610 GOTO 1540
1620 'Speichern - Drucken
1630 CLS:INPUT"drucken 0/1 ",X
1640 INPUT"Wind ",J:WIND J
1650 LPRINT"von ";J;" bis ":J=1
1660 OPEN"O",#2,"CAS0:ADRESS.DAT"
1670 IF X THEN OPEN"O",#1,"COM0:":PRINT#1,CHR$(24);CHR$(27);"!";CHR$(04):WIDTH"C
OM0:",255
1680 FOR I=0 TO IANZ
1690   IF LEN(NAME(I))>=83 AND INSTR(NAME(I),STRING$(70," "))=0 THEN PRINT#2,NAM
E(I),TELE(I):          IF X THEN PRINT#1,USING"### ";J;:PRINT#1,NAME(I),TELE(I):
J=J+1
1700 NEXT I
1710 CLOSE
1720 LPRINT TAPCNT:GOTO 1620
1730 END
```

EDV-Überweisungsvordrucke

Programmiersprache: EBASIC. Gerätekonfiguration: HX-20, Grundversion. Epson-Drucker FX-80. RS-232-Interface 8145, Anschlußkabel # 714.
Speicherplatzbedarf: 4109 Bytes, für Programm 6577 Bytes inkl.Variablen.
Aufgabe dieses Programmes ist es, Überweisungsvordrucke der Banken zu verarbeiten. Diese Vordrucke gibt es bei jeder Bank auch als Endlosformulare, wahlweise mit Durchschlag. Die Bezeichnung hierfür lautet "Überweisungs-Sammelauftrag". Die Endlosformulare sind mit der Traktorführung versehen. Ein zusätzlicher Traktorantrieb ist für den Matrixdrucker FX-80 nicht unbedingt erforderlich, da die Formulare auch über den Antrieb der Gummiwalze verarbeitet werden können. Zusätzlich zur Formularverarbeitung lassen sich mit diesem Programm Dateien anlegen, die Adresse und Bankverbindung verwalten. Kombiniert mit einem Textverarbeitungsprogramm hat man eine leistungsfähige und kostengünstige Kleinanlage, die einen großen Teil kaufmännischer Arbeiten abdecken kann.
Die Anwendung dieses Programmes setzt keine besonderen Programmierkenntnisse voraus. Nur die Adresse des Anwenders mit der Bankverbindung ist einmalig in das Programm selbst einzugeben. Alles andere erfolgt softwaregesteuert im Dialog mit dem Anwender. Die Handhabung wird dadurch übersichtlich und erfordert nach einiger Übung kein Nachschlagen in der Programmbeschreibung.
DAS PROBLEM
Das Ausfüllen von Überweisungsformularen ist reine Routinearbeit. Auf jedem Überweisungsformular ist der Absender und das eingene Konto einzutragen. Die Adressaten sind zum größten Teil auch immmer die gleichen Kunden, Lieferanten, Versicherungsträger, Versorgungsträger, Finanzamt, Kfz.-Werkstatt usw. Dieses Programm bringt hier eine wesentliche Entlastung. Der Absender mit der eigenen Kontoangabe bleibt immer im Programm abgespeichert. Die Adressen werden durch eine Nummer aus der Datei aufgerufen und automatisch formatiert ausgedruckt. Über die Tastatur sind für den Verwendungszweck max. zwei Zeilen plus Überweisungsbetrag einzugeben. Sollte sich an den Adressen etwas ändern, sind Korrekturen jederzeit möglich; die alte Datei wird gelöscht und durch die korrigierte Fassung ersetzt.
PROGRAMM-BESCHREIBUNG
Das Programm wird nach dem Eingeben mit RUN gestartet und setzt mit der DIM-Anweisung in Adressse 50 Platz für 20 Adressen. Abgestimmt hierauf ist die Anweisug CLEAR 2000. In Adresse 60 wird der Zählwerkstellung ein Variablenname zugeordnet, damit die Mikro-Kassette nach jedem Schreib- oder Lesevorgang der Datei an dieser Stelle, vom Programm aus, wieder zurückgespult werden kann. In Adresse 70 bis 90 werden Formate für den späteren PRINT-USING-Befehl festgelegt. Eine Abfrage verzweigt das Programm in die Rubriken "Formularausdruck" und "Adressendatei".
FORMULARAUSDRUCK
Dieser Programmteil deckt den Bereich von Adresse 180 bis Adresse 510 ab. In der ersten Abfrage wird festgelegt, ob die Adresse aus der Datei ausgewählt werden kann oder ob die Adresse über die Tastatur anschließend eingegeben werden soll. Wird eine in der Datei abgespeicherte Adresse verlangt, besorgt das Unterprogramm von Adresse 1700 bis 1770 (GOSUP1700) das Einlesen der gesamten Datei. Nach dem Einlesen wird D = 1 als Flag gesetzt, weil bei weiteren Formularausdrucken nicht jedesmal die ganze Datei neu gelesen werden muß. Erst bei einem neuen Programm-Start wird D = 0 gesetzt und damit das Einlesen der Daten möglich (250). Die Zuteilung der Adressennummer erfolgt in der Druckroutine (370 bis 510) durch die Zuweisung des Variablennamens C. Da C bei der Abfrage in Adresse 210 als String dekla-

riert wurde, muß dieses Zeichen vorher in einen Zahlenwert umgewandelt werden (220). Zwei Zeilen für den Verwendungszweck und der Überweisungsbetrag sind noch einzugeben (310 bis 340). Anschließend wird der Drucker angesteuert, welcher das Formular bedruckt. Nach dem Ausdruck wird automatisch das folgende Formblatt in Position gebracht (Adresse 500).
Soll keine abgespeicherte Adresse verwendet werden, so kann man die Adresse auch direkt eintippen. Hierzu wird das Unterprogramm von Adresse 1610 bis 1690 (GOSUP 1610) aufgerufen. Schritt für Schritt werden dann die erforderlichen Daten abgefragt. Dabei ist die jeweils angegebene Datenlänge in der darüberliegenden Zeile angedeutet. Nach Ablauf dieser Subroutine wird in das Hauptprogramm, Adresse 310, zurückgesprungen. Zwei Zeilen für Verwendungszweck sowie der Überweisungsbetrag sind einzugeben. Der Drucker wird angesteuert und das Formular bedruckt. Nach dem Ausdruck wird automatisch das folgende Formblatt in Position gebacht (Adresse 500). In Adresse 510 wird der Kontakt mit dem Drucker geschlossen und die LCD-Anzeige gelöscht.

ADRESSVERWALTUNG

Dieser Programmteil deckt den Bereich von Adresse 570 bis Adresse 1510 ab. Sieben Routinen und der Rücksprung zur Erstabfrage können durch Eingabe des entsprechenden Buchstabens angesprochen werden.Die einzelnen Routinen sind im Listing durch entsprechende REM-Hinweise verdeutlicht.

DRUCKER-ANPASSUNG

Die Verbindung des HX-20 mit dem EPSON-Drucker FX-80 geschieht mit dem Anschlußkabel #714. Beim Anschluß des Kabels müssen die Geräte ausgeschaltet sein.Im FX-80 muß das RS-232-Interface 8145 installiert sein. Mit dem OPEN-Befehl in Adresse 370 bzw. 1340 wird die Schnittstelle RS 232 angesprochen (COM0) und die Datenübertragungsrate spezifiziert (48N22).
Der Datenübertragungsschlüssel lautet:

```
4 = Baud Rate      : 1200 bps
8 = Wortlänge      : 8 Bit Wort
N = Parity         :  No Parity Check
2 = Stop Bits      : 2 Stp Bits
2 = Aktive         : CTS = aktiv
    Kontroll-        DSR = aktiv
    Leitung          RTS = + aktiv
                     CD  = aktiv
```

Dieser Übertragungsschlüssel weicht von der standardmäßig eingestellten Version ab. Damit die übertragenen Daten auch vom Matrixdrucker FX-80 empfangen werden können, müssen die DIP-Schalter mit dem Datenübertragungsschlüssel aus dem OPEN-Befehl abgestimmt werden (Vgl. FX-80-Handbuch, Seite 40 bis 46).
Folgende DIP-Schalterstellungen sind herzustellen:

```
Grundplatine FX-80   DIP 1:01010101
                     DIP 2:1110
8145-Platine(RS232)  DIP 1:10101111
                     DIP 2:0101
```

1 = ON ; 0 = OFF

DATENEINGABE

Nach dem Einschalten mit RUN bzw. PF5 starten. Folgende Adresse erscheint:

```
ADRESSENDATEI    : A
FORMULARAUSDRUCK
FüR üBERWEISUNG : F
```

Hier wird verzweigt in den Programmteil "Adressverwaltung " oder "Formularausdruck".

ADRESSVERWALTUNG

Nach dem Drücken der Taste A erscheint folgende Anzeige, die jedoch auf der LCD-Anzeige nicht vollständig dargestellt werden kann:

```
ASRESSEN
        ...    EINGEBEN      : E
        ...    KORRIGIEREN   : K
        ...    LÖSCHEN       : L
        ...    SORTIEREN     : S
        ...    ANHÄNGEN      : A
        ...    DRUCKEN       : D
        ...    AUF BAND      : B
      FORMULARAUSDRUCK       : F
```

Die einzelnen Befehle führen folgende Anweisungen aus:
E: Adressenersteingabe. Bereits im Computer enthaltene Adressen werden gelöscht. Eine neue Adressendatei kann angelegt werden. Bereits auf Band abgespeicherte Daten werden nicht gelöscht!.Das Programm fragt nacheinander Anrede, Vorname, Name, Straße, Postleitzahl mit Wohnort, Bankverbindung, Bankleitzahl und Konto-Nr. ab. Natürlich kann für

Anrede auch ein Firmenname und für Straße ein Postfach angegeben werden. Alle Eingaben sind mit RETURN abzuschließen. Außer bei Bankleitzahl und Konto-Nr. dürfen die Eingaben geringfügig länger sein als jeweils angegeben. Auf dem Formular erscheint die tatsächliche Eingabe, während der Ausdruck der Adressendatei nur die angegebenen Wortlängen berücksichtigt. Diese Einschränkung wird deshalb vorgenommen, damit pro Adressat auch nur eine Druckerzeile im Schmalschriftmodus benötigt wird.
Nach Eingabe der Konto-Nr. erscheint die Anzeige:
ADRESSENEINGABEN : ∧
ENDE DER EINGABEN : E
∧: Das Programm fragt eine neue Adresse ab. Die Adressen werden aufsteigend nummeriert.
E: Das Programm springt ins Menü zurück.
Sollte die Adressdatei noch nicht im Computer abgespeichert sein, so erfolgt vor Ausführung der Befehle K, L, S, A, und D automatisch das Einlesen der Adressendatei. Die Mikrokassette ist vor Ausführung dieser Befehle an den Dateianfang zu positionieren. Es erscheint die Anzeige:

DATENSPEICHERUNG
BITTE WARTEN

Anschließend werden diese Befehle automatisch ausgeführt.
K: Korrektur. Sämtlich im Computer befindlichen Adressen, ganz gleich, ob über Tastatureingabe oder Eingabe vom Band, lassen sich durch diese Routine korrigieren. Nach Betätigung der K-Taste erfolgt die Abfrage:

KORREKTUR IN Z.NR.

Die entsprechende Zeilennummer ist einzutippen und mit der RETURN-Taste abzuschließen. Jetzt wird abgefragt:

SPALTEN-NR.

Anrede = 1.Spalte,Vorname=2.Spalte usw. Nachdem die Spalten-Nr. eingegeben und mit RETURN abgeschlossen wird, erscheint der Text auf der LCD-Anzeige. Zur Korrektur ist der Cursor eine Zeile höher in die Textzeile zu setzen. Die Korrektur kann vorgenommen werden, auch unter Anwendung der INSERT-oder DELETE-Taste und ist mit RETURN abzuschließen. Wird gleich nach Erscheinen des zu korrigierenden Textes die RETURN-Taste gedrückt, ist dieser Text gelöscht oder mit Leertasten überschrieben. Das Programm springt ins Menü zurück.
L: Löschen einer kompletten Adresse für die im Computer vorhandene Datei. Nach Betätigung der L-Taste erfolgt die Abfrage:
LÖSCHEN IN Z.NR.
Die entsprechende Zeilennummer ist einzutippen und mit der RETURN-Taste abzuschließen. die komplette Datei ab dieser Zeile wird neu durchnummeriert. Sind mehrere Adressen zu löschen, sollte mit der höchsten Zeilennummer begonnen werden, da alle folgenden Adressennummern um jeweils 1 gekürzt werden und diese gekürzte Adressen-Nummerierung bei der weiteren Bearbeitung zu berücksichtigen ist. Es kann jeweils nur eine Adresse gelöscht werden, anschließend springt das Programm ins Menü zurück.
S: Sortieren sämtlicher Adressen für die im Computer vorhandene Datei. Nach Betätigung der S-Taste erfolgt der Ausdruck "SORTIERROUTINE LÄUFT" auf der LCD-Anzeige. Sortiert wird alphabetisch nach Spalte 3 (Name). Diese Durchnummerierung erfolgt ebenfalls neu, so daß hier die Möglichkeit besteht, auch die später erfolgten zusätzlichen Adresseneingaben (ADRESSEN ANHÄNGEN: A) alphabetisch einzugliedern. Nach beendigter Sortierroutine springt das Programm ins Menü zurück.
A: Anhängen weiterer Adressen zu der bereits im Computer vorhandenen Datei. Nach Betätigung der A-Taste fragt das Programm nacheinander Anrede, Vorname, Name, Straße ab (vgl. Adressenersteingabe).
D: Druckt die aktuelle im Computer vorhandene Datei im Schmalschriftmodus mit laufender Adressen-Nummerierung aus. Die Anwendung dieser Routine sollte nach jeder beendeten Dateiänderung erfolgen, da nur die aktuelle Datei für weitere Arbeitsvorgänge herangezogen wird.
B: Speichert die im Computer vorhandene Datei auf Mikro-Kassette ab. Jedoch Vorsicht bei neu ange-

legten Dateien. Da nach jedem Schreib- und Lesevorgang die Mikro-Kassette automatisch an den Dateianfang gespult wird, würde eine neue Datei die vorhandene Datei überschreiben. Handelt es sich jedoch um eine Änderung der alten Datei, dann muß die alte Datei durch die aktuelle Datei überschrieben werden, und für diesen Fall, der zweifellos häufiger eintritt, wäre die Mikro-Kassette richtig positioniert.
F: Das Programm springt in den Teil "Formularausdruck". Soll das Programm abgebrochen werden, ist die BREAK-Taste zu betätigen. Ein neuer Start kann dann wieder mit RUN aufgerufen werden. Nach diesem Vorgang sind alle Variablen einschließlich der Datei gelöscht.
FORMULARAUSDRUCK
Nach dem Drücken der Taste F erscheint die Anzeige:

ADRESSEN WÄHLEN: 1...
TASTATUREINGABE: T

Soll eine Adresse aus der Datei gewählt werden, so ist die Adressen-Nr. einzugeben und mit RETURN abzuschließen. Bevor die RETURN-Taste gedrückt wird, sollte man sich vergewissern, ob die Mikro-Kassette mit der Datei auch richtig positioniert ist. Sofern die Datei nicht schon vorher aufgerufen wurde, wird die gesamte Datei vom Band eingelesen. Folgende Anzeige erscheint:

GESPEICHERTE ADRESSE
WIRD ZUGETEILT

Nach der Datenübertragung wird das Band wieder an den Dateianfang zurückgespult.
Folgende Anzeige erscheint:
VERWENDUNGSZWECK 1.Z
v.....v
Der Text für die erste Zeile kann eingegeben werden. Die dargestellte Zeilenlänge ist zu beachten. Mit RETURN abschließen .
Folgende Anzeige erscheint:
VERWENDUNGSZWECK 2. Z
v.....v
wie zuvor, mit RETURN abschließen.
Folgende Anzeige erscheint:

ÜBERWEISUNGSBETRAG

Der Betrag ist mit Dezimalpunkt und zwei Stellen hinter dem Komma einzugeben, z.B.: 349.00.Mit RETURN abschließen. Der Formularausdruck wird ausgeführt. Folgende Anzeige erscheint:

FORMULARAUSDRUCK
BITTE WARTEN

dann: WEITER AUSDRUCKE
(J/N)

N: Programmende. Alle Variablen werden gelöscht einschließlich der Datei. Neustart mit RUN möglich.
J: Rücksprung ins Hauptmenü. Alle Variablen inklusive der Datei bleiben für weiter Verwendung erhalten.
EINGABEFEHLER
Solange eine Eingabe noch nicht mit RETURN abgeschlossen ist, besteht die Möglichkeit zur Korrektur in dieser Zeile:
Löschen, Einsetzten, Ersetzen.
Bei allen Eingaben können auch Sonderzeichen sowie Punkte oder Kommata verwendet werden. Nach Abschluß einer Eingabe mit RETURN besteht für den Programmteil "Adressverwaltung" die Möglichkeit über die Routine "ADRESSE KORRIGIEREN: K" Änderungen vorzunehmen. Im Programmteil "Formularausdruck" ist keine Korrektur-Routine vorgesehen. Hier muß das Programm mit BREAK abgebrochen und mit RUN neu gestartet werden. Bei Abfragen, die nicht mit RETURN bestätigt werden müssen, sind große Buchstaben einzugeben. Andere Zeichen als die jeweils vorgesehenen werden ignoriert.
PROGRAMMGRENZEN
Die Anzahl der abzuspeichernden Adressen ist vom Programm her auf 20 Stück begrenzt(50). Die CLEAR-Anweisung ist hierauf mit dem Faktor 100 auf den Wert 2000 abgestimmt (40). Für größere Dateien sind diese beiden Werte entsprechend abzuändern. Da bei dieser Datei 8 Spalten angelegt werden, würde der Ausdruck DIM A$ (20,8) genügen. Die 9. Spalte ist für die Telefonnummer reserviert, die jedoch für diesses Programmm nicht benötigt wird. Die maximale Größe der Adressendatei wird durch den Speicherplatz des Computers bestimmt.
VARIABLEN-NAMEN
Strings:
A$ = Adresse; B$ = 1. Abfrage (80-90); C$ = 2. Abfrage (120-130); D$ = Verwendungszweck 1.Zeile; E$ =

Verwendungszweck 2.Zeile; F$ = Überweisungsbetrag; H$ = 3.Abfrage (620); I$ = 4.Abfrage (1100); J$ = 5. Abfrage (1840); K$ = 6. Abfrage (2140); U2$ ... U4$ = Usigformat.
Real-Integer:
A = TAPCNT; B = Zähler für Dateiangabe; C = VAL (C$); D = Flag; E = äußere Schleife; F = innere Schleife; G = Korrekturzeile; H = Korrekturspalte; I = Flag; Z = Schleife Adressenlisting.
BEISPIEL: FORMULARAUSDRUCK
Das Programm mit RUN starten. Gewünschten Programmteil wählen: F. Adresseneingabe über Tastatur: T. Anrede eingeben, mit RETURN bestätigen. Vorname eingeben, mit RETURN bestätigen. Name eingeben, mit RETURN bestätigen. Straße eingeben, mit RETURN bestätigen. PLZ mit Wohnort eingeben, mit RETURN bestätigen. Bankverbindung eingeben, mit RETURN bestätigen. Bankleitzahl eingeben, mit RETURN bestätigen. Konto.-Nr. eingeben, mit RETURN bestätigen. Verwendungszweck 1.Zeile eingeben, mit RETURN bestätigen. Verwendungszweck 2. Zeile eingeben, mit RETURN bestätigen. Überweisungsbetrag eingeben, mit RETURN bestätigen. Die vorgegebenen Wortlängen sind bei Bankleitzahl und Konto-Nr. unbedingt einzuhalten. Alle anderen Eingaben können geringfügig länger ausfallen. Wie ein Formularausdruck über den angeschlossenen Drucker aussehen kann, zeigt das abgedruckte Beispiel. Der Computer steht für weitere Aufgaben bereit. Für weitere Aufgaben zurück ins Hauptmenü: J (Unterbrechung durch BREAK und Neustart durch RUN ist nicht erforderlich).
BEISPIEL: ADRESSENDATEN
Programmstart mit RUN nur erforderlich, wenn zuvor nicht bereits ins Hauptmenü zurückgesprungen wurde. Gewünschten Programmteil wählen: A. Für die Adresseneingabe (Adressenersteingabe!) E. Anrede eingeben, mit RETURN bestätigen. Ende der Eingabe: E. Das Programm bleibt im Programmteil "Adressendatei". Weitere Befehle können vorgenommen werden. Durch Drücken von "F" gelangt man wieder in den Programmteil "Formularausdruck". Soll das Programm beendet werden: BREAK-Taste drücken. Alle Variablen sind dann im Programmteil gelöscht.

Arno W.Wagner

```
1  'EDV-Ü-D
2  'für Chip-Special
3  'von Arno W. Wagner
20 REM SPEICHERPLATZ FÜR PROGRAMM
:4103 BYTES
30 REM SPEICHERPLATZ INCL.VARIABLEN
:6571 BYTES
40 CLEAR 2000
50 DIM A$(20,9)
60 A=TAPCNT
70 U2$="ö          ö"
80 U3$="ö                ö"
90 U4$="ö                    ö"
100 CLS
110 PRINT"ADRESSENDATEI     :A"
120 PRINT"FORMULARAUSDRUCK"
130 PRINT"FÜR ÜBERWEISUNG   :F"
140 B$=INKEY$
150 IFB$="A"THEN570
160 IFB$="F"THEN180
170 GOTO 140
180 CLS
190 PRINT"ADRESSEN WAHLEN:1..."
200 PRINT"TASTATUREINGABE:  T "
210 INPUT C$
220 C=VAL(C$)
230 IFC$="T"THEN300
240 IFC$="0"THEN180
250 CLS:IFD=1THEN310
260 PRINT"GESPEICHERTE ADRESSE"
270 PRINT"WIRD ZUGETEILT":PRINT
280 GOSUB1700
290 GOTO310
300 C=0:GOSUB1610
310 PRINT"VERWENDUNGSZWECK 1.Z":PRI
NT"v.....
.....v":LINE INPUT D$
320 PRINT"VERWENDUNGSZWECK 2.Z":PRI
NT"v.....
.....v":LINE INPUT E$
330 PRINT"ÜBERWEISUNGSBETRAG":LINE
INPUT F$
340 CLS
350 PRINT"FORMULARAUSDRUCK"
360 PRINT"BITTE WARTEN"
370 OPEN"O",#1,"COM0:(48N22)"
380 WIDTH"COM0:",255
390 PRINT#1,CHR$(27);"C";CHR$(25);
400 PRINT#1,"          ";A$(C,1);" "
:A$(C,2);" ";A$(C,3)
```

```
410 PRINT#1,"          ";A$(C,4);"
":A$(C,5)::PRINT#1,CHR$(27);"D";CHR
$(52);CHR$(0);CHR$(9);:PRINT#1,USIN
G U2$;A$(C,7)
420 PRINT#1
430 PRINT#1,"          ";:PRINT#1,US
ING U2$;A$(C,8);:PRINT#1,"     ";A$(
C,6)
440 PRINT#1:PRINT#1:PRINT#1
450 PRINT#1,"          ";D$
460 PRINT#1,"          ";E$;:PRINT#1
,CHR$(27);"D";CHR$(52);CHR$(0);CHR$
(9);:PRINT#1,"***";F$
470 PRINT#1
480 PRINT#1:PRINT#1:PRINT#1:PRINT#1
490 PRINT#1,"          000232122
 A.W.WAGNER,KAISER-WILHELM-PL.5, 71
00 HN"
500 PRINT#1,CHR$(12);
510 CLOSE#1:CLS
520 PRINT"WEITERE AUSDRUCKE":PRINT"
( J/N ) ? "
530 H$=INKEY$
540 IF H$="J"THEN100
550 IF H$="N"THENEND
560 GOTO530
570 CLS
580 PRINT"ADRESSEN EINGEBEN :E"
590 PRINT"...   KORRIGIEREN :K"
600 PRINT"...      LöSCHEN :L"
610 PRINT"...    SORTIEREN :S"
620 PRINT"...     ANHÄNGEN :A"
630 PRINT"...      DRUCKEN :D"
640 PRINT"...     AUF BAND :B"
650 PRINT"FORMULARAUSDRUCK  :F"
660 I$=INKEY$
670 IFI$="E"THEN1520
680 IFI$="K"THEN860
690 IFI$="L"THEN860
700 IFI$="S"THEN860
710 IFI$="A"THEN860
720 IFI$="D"THEN860
730 IFI$="B"THEN760
740 IFI$="F"THEN180
750 GOTO660
760 PRINT:PRINT"DATENSPEICHERUNG"
770 PRINT"BITTE WARTEN"
780 OPEN"O",#1,"CAS0:DAT"
790 PRINT#1,B
800 FORE=1TOB
810 FORF=1TO9
820 PRINT#1,A$(E,F)
830 NEXTF
840 NEXTE
850 CLOSE#1:WIND A:GOTO1510
860 CLS
870 IFD=1THEN920
880 PRINT"EINLESEN DER DATEN"
890 PRINT"BITTE WARTEN"
900 GOSUB 1700
910 CLS
920 REM ********** K O R R E K T U
R **********
930 IFI$<>"K"THEN990
940 PRINT"KORREKTUR IN Z.NR.":INPUT
 G
950 PRINT"SPALTEN-NR.":INPUT H
960 PRINT A$(G,H)
970 LINE INPUT A$(G,H)
980 CLS
990 IFI$<>"L"THEN1090
1000 REM ********** L ö S C H E N *
*********
1010 PRINT"LöSCHEN IN Z.NR.":INPUT
G
1020 FORE=G TO B
1030 FORF=1TO9
1040 A$(E,F)=A$((E+1),F)
1050 NEXTF
1060 NEXTE
1070 B=B-1
1080 CLS
1090 IFI$<>"S"THEN1210
1100 REM ********** S O R T I E R E
 N **********
1110 PRINT"SORTIERROUTINE LÄUFT"
1120 PRINT"BITTE WARTEN"
1130 I=0
1140 FORE=1TO(B-1)
1150 IFA$(E,3)<=A$((E+1),3)THEN1180
1160 FORF=1TO9:SWAPA$(E,F),A$((E+1)
,F):NEXTF
1170 I=1
1180 NEXTE
1190 IFI=1THEN1130
1200 CLS
1210 IFI$<>"A"THEN1310
1220 REM ********** A N H Ä N G E N
 **********
1230 C=B+1:GOSUB1610:B=C
1240 PRINT"WEITERE":PRINT"ADRESSENE
INGABEN  :^"
1250 PRINT"ENDE DER EINGABEN :E"
1260 J$=INKEY$
1270 IFJ$="^"THEN1230
1280 IFJ$="E"THEN1300
1290 GOTO1260
1300 CLS
1310 IFI$<>"D"THEN1510
1320 REM ********** D R U C K E N *
*********
1330 PRINT"AUSGABE ÜBER DRUCKER":PR
INT"BITTE WARTEN"
1340 OPEN"O",#1,"COM0:(48N22)"
1350 WIDTH"COM0:",255
1360 PRINT#1,CHR$(15);
```

```
1370 PRINT#1,CHR$(27);"l";CHR$(5);
1380 FOR Z=1 TO B
1390 PRINT#1,USING"###";Z;:PRINT#1,
" ";
1400 PRINT#1,USING U3$;A$(Z,1);:PRI
NT#1," ";
1410 PRINT#1,USING U2$;A$(Z,2);:PRI
NT#1," ";
1420 PRINT#1,USING U3$;A$(Z,3);:PRI
NT#1," ";
1430 PRINT#1,USING U4$;A$(Z,4);:PRI
NT#1," ";
1440 PRINT#1,USING U4$;A$(Z,5);:PRI
NT#1," ";
1450 PRINT#1,USING U4$;A$(Z,6);:PRI
NT#1," ";
1460 PRINT#1,USING U2$;A$(Z,7);:PRI
NT#1,"  ";
1470 PRINT#1.USING U2$;A$(Z,8)
1480 NEXTZ
1490 PRINT#1.CHR$(12);CHR$(27);"§";
1500 CLOSE#1
1510 GOTO580
1520 REM ********** A D R E S S E N
 E I N G A B E **********
1530 C=0:D=1
1540 C=C+1:PRINT"ADRESSE ";C:GOSUB1
610:B=C
1550 PRINT"WEITERE":PRINT"ADRESSENE
INGABEN  :^"
1560 PRINT"ENDE DER EINGABEN :E"
1570 K$=INKEY$
1580 IFK$="^"THEN1540
1590 IFK$="E"THEN580
1600 GOTO1570
1610 PRINT"ANREDE":PRINT"v---------
----v":LINE INPUT A$(C,1)
1620 PRINT"VORNAME":PRINT"v--------
v":LINE INPUT A$(C,2)
1630 PRINT"NAME":PRINT"v-----------
--v":LINE INPUT A$(C,3)
1640 PRINT"STRAßE":PRINT"v---------
---------v":LINE INPUT A$(C,4)
1650 PRINT"PLZ ORT":PRINT"v--------
----------v":LINE INPUT A$(C,5)
1660 PRINT"BANKVERBINDUNG":PRINT"v-
-----------------v":LINE INPUT A$(C
,6)
1670 PRINT"BANKLEITZAHL":PRINT"v---
------v":LINE INPUT A$(C,7)
1680 PRINT"KONTO-NR.":PRINT"v------
--v":LINE INPUT A$(C,8)
1690 CLS:RETURN
1700 OPEN"I",#1,"CAS0:DAT"
1710 INPUT#1,B
1720 FORE=1TOB
1730 FORF=1TO9
1740 INPUT#1,A$(E,F)
1750 NEXTF
1760 NEXTE
1770 CLOSE#1:WIND A:D=1:RETURN
```

Ladenkasse

Programmiersprache: EBASIC. Gerätekonfiguration: HX-20, Grundversion. TF-20-Diskettenstation.
Speicherplatzbedarf: 3996 Bytes.
Das Erstellen von Kassenquittungen mittels eines Hand-Held-Computers ermöglicht dieses Programm. Ein externer Drucker ist nicht notwendig. Bei knappem Speicherplatzbedarf kann das Programm durch Weglassen von Kommentaren auf etwa 2500 Bytes gekürzt werden. Die zu bearbeitenden Artikel/Leistungen sind in einer RAM-Datei abgelegt, die für 240 Artikel etwa 4400 Bytes reserviert.

PROGRAMMBESCHREIBUNG

Sobald das Programm eingegeben und gestartet ist, meldet sich der Rechner akustisch und zeigt das Programm-Menü an. Man kann nun durch einen kurzen Druck auf die Taste mit der Ziffer "8" das Beispiel-RAM mit einigen Artikeln einlesen. Der HX-20 fragt dazu erst ab, ob er das im Speicher vorhandene RAM-File löschen soll. Nach einem Druck auf die "J"-Taste vergewissert er sich noch einmal, ob er wirklich alles löschen soll. Er erfragt dann die Bandposition, ab der das RAM-File auf der Mikro-Kassette gespeichert ist. Bei Verwendung der TF-20-Disketten-Station ist nur ein Druck auf die RETURN-Taste erforderlich. Wird die Frage "RAM löschen?" nicht mit "J" (sondern mit einem beliebigen anderen Zeichen) beantwortet, so wird das zu ladende RAM-File an das vorhandene angeknüpft, so daß dann beide Inhalte gleichzeitig zur Verfügung

stehen. Nach dem Laden meldet sich der Rechner mit dem Programm-Menü zurück, in das man auch jederzeit durch Betätigen der BREAK-Taste und Drücken der Taste PF1 zurückkehren kann.
Menü-Teil "1" ermöglicht den Ausdruck der im RAM-File gespeicherten Artikel. Als Augabeeinheiten stehen der Bildschirm, der eingebaute Minidrucker und jeder externe Drukker zur Verfügung. Die Daten des RAM-Files können nun verändert ("3"), gespeichert ("4"), erweitert ("6") oder gelöscht ("7") werden.
QUITTUNG ERSTELLEN
Das Erstellen eines Quittungsbeleges wird mit dem Menü-Teil "5" eingeleitet, wobei der Kopf beliebig gestaltet werden kann. Nach Eingabe der Artikel-Nr. (z.B. "201"; ohne RETURN - kürzere Artikel-Nummern durch Leerzeichen auffüllen) wird auf dem LCD-Display der zugehörige Text angezeigt, vor dem ein Sonder-Cursor blinkt. Der HX-20 wartet blinkend auf ein Zeichen und verzweigt dann wie folgt: Bei "+" wird die im Bild gezeigte Zeile gedruckt. Bei "-" wird mit gleichem Text der Betrag negativ ausgegegben (als Gutschrift). Bei "*" wird ein Faktor erfragt (siehe: 2 x Buchhaltung). Und bei allen anderen Eingaben nimmt der HX-20 an, daß Sie sich bei der Artikel-Nummer geirrt haben und erfragt die nächste Position.
Wird bei der Frage nach einer Artikel-Nummer ein "+" eingegeben, so wird die Summe der bereits gedruckten Positionen ausgegeben. Ein Doppelkreuz schließt die Belegausgabe ab (einschließlich Ausdruck der Mehrwertsteuer) und zeigt das Programm-Menü an. Wird eine unbekannte Artikel-Nummer eingegeben (z.B. "0"), so kann ein beliebiger Text mit beliebigem Preis gedruckt werden (beides wird vom Programm erfragt; wird nach der Frage "Text ?" nur die RETURN-Taste gedrückt, vermutet der HX-20 einen Irrtum und erfragt die nächste Artikel-Nummer).
Die Eingabe eines %-Zeichens an dieser Stelle ermöglicht die Ausgabe von MWST.-, Skonto- oder Rabattzeilen, wobei der Bezugsbetrag jeweils die letzte Zeile des Ausdrucks ist. Sie können also Rabatt auf einzelne Positionen oder Skonto auf die Summe getrennt drucken. Das Vorzeichen der erfragten %-Zahl (der Bezugsbetrag wird angezeigt) bestimmt, ob es sich um einen Aufschlag oder einen Nachlaß handelt. Es können jederzeit Zwischensummen gebildet werden. Wird auf die Frage nach der Positions-Nummer die Taste "S" gedrückt, so gibt der HX-20 die Summe der Tages-Umsätze aus. Durch die Funktionstaste PF3 wird eine Gutschrift vorbereitet und PF4 berechnet 14% MWST. zur letzten Zeile hinzu.
Das Programm verbleibt ebenso wie die RAM-Datei auch nach dem Abschalten des HX-20 im Speicher und steht nach dem Einschalten sofort zur Verfügung (über das HX-20-Menü). Wurde der Rechner nicht zwischendurch für andere Zwecke benutzt, stehen alle Artikel ohne ein Nachladen vom Band sofort zur Verfügung. Bei der Belegausgabe mit "5" wird allerdings erst einmal die Summe aller Belege des Vortages ausgegeben!

Jürgen und Susanne Miersch

```
1    'LADENKASSE
2    'für Chip-Special
3    'von J.& S. Miersch
110 CLEAR 500,5302: WIDTH 20,255: D
EFFIL 22,0: DEFINT I-N: DF$="COM0:(
68N27)": GOSUB 830
120 KEY1,CHR$(12)+"GOTO160"+CHR$(13
): KEY4,"%14"+CHR$(13)+"MwSt."+CHR$
(13): KEY3,"0  Gutschrift"+CHR$(13)
130 D$=DATE$: D$=MID$(D$,4,2)+"."+L
EFT$(D$,2)+"."+RIGHT$(D$,2)
140  TITLE"Quit": POKE&H169,4: ON E
RROR GOTO 1010
150 '***  MENUE  ***
160 GOSUB 990: PRINT"1-Druck   2-EN
DE  3-Korr.   4-Sichern 5-Beleg
 6-Insert  7-Delete  8-Laden  ";
170 ON VAL(INPUT$(1)) GOTO 190,1030
,310,350,400,630,680,720: GOTO 160
180 '***  DRUCKEN  ***
190 CLS: PRINT"0-Bildschirm": PRINT
"1-Minidrucker": PRINT"2-Großdrucke
```

```
r": K=VAL(INPUT$(1))
200 IF K=0 THEN V24$="SCRN:" ELSE I
F K=1 THEN V24$="LPT0:" ELSE IF K=2
 THEN V24$=DF$ ELSE 190
210 CLOSE#2: IF K=0 THEN K=9 ELSE K
=13
220 OPEN"O",#2,V24$: PRINT#2,"Nr. B
ez.";TAB(K+9);"DM";CHR$(10)
230 FOR I=1 TO 240
240     GET%I,NR,DM,A$: IF NR=999 T
HEN 280 ELSE CLS
250     PRINT#2,USING"### ";NR;
260     PRINT#2,LEFT$(A$,K);: PRINT
#2,USING"####.##";DM
270 NEXT
280 PRINT#2," ": CLOSE#2: GOTO 160
290 '
300 '***  KORRIGIEREN  ***
310 CLS: PRINT"KORRIGIEREN": PRINT:
 INPUT"Artikel-Nr. ";K: GOSUB 860
320 IF K=0 THEN 160 ELSE GOSUB 900:
 GOTO 310
330 '
340 '***  RAM AUF MC/DISK SICHERN
***
350 CLS: INPUT"WIND ";W$: IF W$=""
THEN 360 ELSE WIND: WIND VAL(W$): L
PRINT"RAM auf MC ab";TAPCNT
360 OPEN"O",#2,"RAM"
370 FOR I=0 TO LAST: GET%I,NR,DM,A$
: PRINT#2,NR,DM,A$: NEXT: CLOSE#2:
GOTO 160
380 '
390 '***  BELEG ERSTELLEN  ***
400 CLS: S=0: GOTO 410: INPUT"Name
";N$: 'NAME NUR FÜR BUCHUNGSSATZ
410 GET%0,NR,DM,A$: A$=LEFT$(A$,8):
 IF A$=D$ THEN 430
420 LPRINT A$;" insg. DM";: LPRINTU
SING"####.##";DM: S=DM: DM=0: PUT%0
,NR,DM,D$: GOTO 590
430 LPRINT"HX-20 Club       ";D$;"
 c/o Jürgen Miersch          824 1
6 48": LPRINT: LPRINT
440 CLS: PRINT"Pos. ? ";: X$=INPUT$
(1): PRINT X$;: IF INSTR("=+%#sS",X
$)=0 THEN 460
450 ON INSTR("=+%#sS",X$) GOTO 550,
550,520,570,600,600: GOTO 510
460 X2$=INPUT$(2): PRINT X2$: K=VAL
(X$+X2$): GOSUB 860: IF K=0 THEN 51
0 ELSE GET%K,NR,DM,A$
470 PRINT: PRINT"  ";A$;: L=CSRLIN:
 GOSUB 970: ON INSTR("+-*",O$) GOTO
 560,500,480: GOTO 440
480 PRINTCHR$(26);: INPUT"FAKTOR ";
N: LPRINT MID$(STR$(N),2);" x": LPR
INT A$;: LPRINT USING" ####.##";DM
490 DM=N*DM: A$="                =":
 GOTO 560
500 DM=-DM: GOTO 560
510 INPUT"Text ";A$: IF A$="" THEN
440 ELSE A$=LEFT$(A$+SPACE$(16),16)
: INPUT"DM    ";DM: GOTO 560
520 PRINT"Wieviel Prozent von DM";D
M;: INPUT" ";P: DM=P/100*DM
530 INPUT"Bez. ";A$: A$=LEFT$(STR$(
P)+"% "+A$+SPACE$(16),16)
540 IF P>0 THEN MID$(A$,1,1)="+": G
OTO 560 ELSE 560
550 A$="Summe          DM": DM=S: S=
0: LPRINT STRING$(24,"_"): GOTO 560
560 S=S+DM: LPRINT A$;: LPRINT USIN
G" ####.##";DM: LPRINT: GOTO 440
570 GET%0,NR,DM: DM=DM+S: PUT%0,NR,
DM
580 LPRINT"incl. 14% MwSt.  ";: LPR
INTUSING"####.##";S/1.14*0.14
590 LPRINT STRING$(24,"="); STRING$
(3,10): GOTO 160
600 GET%0,NR,DM,A$: A$=LEFT$(A$,8):
 GOTO 420
610 '
620 '***  RAM-FILE ERWEITERN  ***
630 CLS: INPUT"Vor Pos ";K: IF K=0
THEN 160 ELSE GOSUB 860: IF K=0 THE
N 160
640 FOR I=LAST TO K STEP-1: GET%I,N
R,DM,A$: PUT%I+1,NR,DM,A$: NEXT
650 CLS: GOSUB 910: LAST=LAST+1: LP
RINT USING"### ";NR;: LPRINT LEFT$(
A$+SPACE$(12),12);: LPRINT USING"##
##.##";DM: KEY5,STR$(K+1)+CHR$(13):
 GOTO 630
660 '
670 '***  RAM-FILE BEREINIGEN  ***
680 CLS: INPUT"Welche  Position
löschen ";K: IF K=0 THEN 160 ELSE G
OSUB 860: IF K=0 THEN 680
690 FOR I=K TO LAST: GET%I+1,NR,DM,
A$: PUT%I,NR,DM,A$: NEXT: LAST=LAST
-1: GOTO 680
700 '
710 '***  RAM VON MC/DISK EINLESEN
 ***
720 PRINT"RAM löschen ?": IF INPUT$
(1)="j" THEN CLS: GOSUB 790: K=0 EL
SE K=LAST+1
730 INPUT"WIND ";W$: IF W$="" THEN
740 ELSE WIND VAL(W$)
740 OPEN"I",#1,"RAM": IF K THEN INP
UT#1,NR,DM: LINE INPUT#1,A$
750 FOR I=K TO 240: IF EOF(1) THEN
760 ELSE INPUT#1,NR,DM: LINE INPUT#
```

```
1,A$: PUT%I,NR.DM,A$: NEXT
760 CLOSE#1: GOSUB 830: GOTO 160
770 '
780 'INITIALISIEREN
790 PRINT"Alles löschen ?": IF INPU
T$(1)<>"j" THEN 160 ELSE CLS: PRINT
"OKAY"
800 NR=999: DM=0: A$=SPACE$(16): FO
R I=1 TO 240: PUT%I,NR,DM,A$: NEXT:
 RETURN
810 '
820 'LAST SUCHEN
830 FOR I=1 TO 240: GET%I,NR: IF NR
=999 THEN LAST=I-1: RETURN ELSE NEX
T: RETURN
840 '
850 'ARTIKEL-NR. SUCHEN
860 FOR I=1 TO LAST: GET%I,NR: IF N
R=K THEN K=I: RETURN ELSE NEXT
870 PRINT"Pos. unbekannt !": SOUND
14,10: SOUND 0,1: K=0: RETURN
880 '
890 'ZEILE EINLESEN
900 CLS: GET%K,NK,DK,AK$: PRINT AK$
: PRINT
910 INPUT"Pos.-Nr. ";X$: IF X$="" T
HEN NR=NK ELSE NR=VAL(X$)
920 INPUT"Preis ";X$: IF X$="" THEN
 DM=DK ELSE DM=VAL(X$)
930 INPUT"Bezeichnung ";X$: IF X$="
" THEN A$=AK$: GOTO 950
940 IF LEN(X$)>16 THEN PRINT"MAX. 1
6 ZEICHEN": GOTO 930 ELSE A$=X$
950 PUT%K,NR,DM,A$: RETURN
960 '
970 LOCATE 0,L: PRINT"
":: LOCATE 0
,L: PRINT"
": LOCATE 0,L: PRINT" ";
: O$=INKEY$: IF O$="" THEN 970 ELSE
 RETURN
980 '
990 SOUND 1,1: SOUND 0,1: SOUND 1,1
: RETURN
1000 '
1010 LPRINT"ERROR": RESUME 160
1020 '
1030 TITLE"": KEY1,"": KEY3,"": KEY
4,"": KEY5,""
```

Schuldtilgung

Programmiersprache: EBASIC.Gerätekonfiguration: HX 20, Grundversion. Speicherplatzbedarf: 1555 Bytes. Eine wichtige Rolle bei der Finanzierung von Innovationen im modernen Marketing spielt das Problem : Kapital aufnehmen, in gleichbleibenden Raten verzinsen und innerhalb einer bestimmten Frist zurückzahlen. Der HX-20 weiß die Antwort, wenn man ihn entsprechend füttert. Das folgende Programm läßt den Anwender in Menü-Technik wählen, welchen der vier Parameter man in Kenntnis der übrigen ausrechnen lassen will: Kapital, Annuität, Zinssatz oder Anzahl Jahre. Mit Ausnahme der Berechnung des Zinssatzes gelingen die übrigen Rechnungen durch Umstellen der Grundformel:

A = K * p(1+1/((1+p) -1))

Einzig beim Zinssatz "p" will uns das nicht recht gelingen. Wenn die analytische Lösung unmöglich ist (transzendente Gleichung) oder wenn sie uns nicht gleich einfällt,so hilft probieren. Der Computer sucht so lange nach dem besseren Wert, bis die Differenz zwischen zwei Durchgängen kleiner als 10- wird (Zeile 960). Dann bricht die Logik ab und wirft den Prozentsatz aus. Ansonsten gibt es keine speziellen Probleme.

Pierre Weber

Wenn Sie dann so weit sind, daß Sie ähnlich gute oder vielleicht noch bessere Programme schreiben können, dann hat sich Ihr Computer bereits amortisiert.
Denn: Die Redaktion CHIP-Special kauft sämtliche Programme auf, die unseren Kriterien standhalten.

Wir halten Ihnen die Daumen!

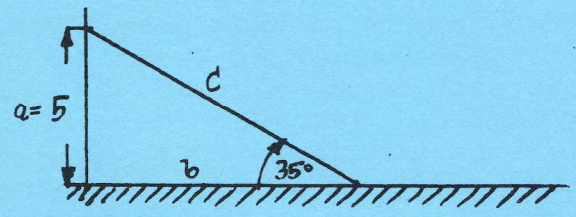
Beispiel

```
          M E N U
1 ANNUITÄT   4 %-SATZ
2 SCHULD     5 END
3 JAHRE
ANNUITÄTEN-BERECHNUNG
-------------------------
SCHULD        : 500000.00
PROZENTSATZ :       6.75
ANZAHL JAHRE:          3
ANNUITÄT      = 189656.57
-------------------------

SCHULD        : 500000.00
 1 .ZINS =      33750.00
TILGUNG   =     155906.57
REST      =     344093.43
-------------------------
SCHULD        : 344093.00
 2 .ZINS =      23226.31
TILGUNG   =     166430.26
REST      =     177663.17
-------------------------
SCHULD        : 177663.00
 3 .ZINS =      11992.26
TILGUNG   =     177664.31
REST      =        -1.14
-------------------------



ANNUITÄT      = 189656.57
PROZENTSATZ :       6.75
ANZAHL JAHRE:          3
SCHULD        : 500000.00
-------------------------
SCHULD        : 500000.00
PROZENTSATZ :       6.75
ANNUITÄT      = 189654.57
ANZAHL JAHRE:          3
-------------------------
SCHULD        : 500000.00
ANZAHL JAHRE:          3
ANNUITÄT      = 189656.57
PROZENTSATZ :       6.75
-------------------------
```

```
1  'Schuldtilgung
2  'für Chip-Special
3  'von Pierre Weber
11 DEFDBL A,K,T,Z
12 U$="------------------------":CLS
13 PRINT"      M E N U    ":SOUND 5,
6
14 PRINT"1 ANNUITÄT   4 %-SATZ"
15 PRINT"2 SCHULD     5 END"
16 PRINT"3 JAHRE";
17 A$=INKEY$:IF A$="" THEN 17
18 ON VAL(A$) GOTO 27,43,47,51,57
19 CLS:SOUND 10,2:INPUT"KAPITAL  ":
K:RETURN
20 LPRINT USING"SCHULD        :######
#";K;:LPRINT".00":PRINT USING"SCHUL
D   :########";K:RETURN
21 LPRINT USING"PROZENTSATZ :      #
#.##";P*100:PRINT USING"PROZENTSATZ
 :   ##.##";P*100:RETURN
22 LPRINT USING"ANZAHL JAHRE:      #
#";N:PRINT USING"ANZAHL JAHRE:   ##"
;N:RETURN
23 LPRINT USING"ANNUITÄT     =#######
#.##";AN:PRINT USING"ANNUITÄT =####
###.##";AN:RETURN
24 SOUND 10,2:INPUT"ZINSSATZ %  ";P
:P=P/100:RETURN
25 SOUND 10,2:INPUT"ANZAHL JAHRE";N
:RETURN
26 SOUND 10,2:INPUT"ANNUITÄT";AN:RE
TURN
27 'Berechnung Annuitaet
28 GOSUB 19:GOSUB 24:GOSUB 25
29 AN=K*(P/(1-(1+P)^-N))
30 LPRINT"ANNUITÄTEN-BERECHNUNG":LP
RINT U$
31 GOSUB 20:GOSUB 21:GOSUB 22:GOSUB
 23:LPRINT U$:LPRINT
32 PRINT"PROTOKOLL (J/N)"
33 A$=INKEY$:IF A$="" THEN 33
34 IF A$<>"J" THEN 13 ELSE 35
35 'Berechnung Jahreszahlen
36 FOR I=1 TO N
37 GOSUB 20:Z=P*K
38 LPRINT I;".";:LPRINT USING"ZINS
=    ########.##":Z:T=AN-Z
39 LPRINT USING"TILGUNG   =    ######
#.##";T:K=K-T
40 LPRINT:LPRINT USING"REST      =
  ########.##";K:LPRINT U$
41 NEXT
42 LPRINT:LPRINT:LPRINT:GOTO 13
43 'Berechnung Schuld
44 CLS:GOSUB 26:GOSUB 24:GOSUB 25
45 K=AN/P*(1-(1+P)^-N):K=INT(K*10+.
5)/10
46 GOSUB 23:GOSUB 21:GOSUB 22:GOSUB
 20:LPRINT U$:GOTO 13
47 'Berechnung Anz.Jahre
48 GOSUB 19:GOSUB 24:GOSUB 26
49 N=LOG(AN/(AN-P*K))/(LOG(1+P))
50 GOSUB 20:GOSUB 21:GOSUB 23:GOSUB
 22:LPRINT U$:GOTO 13
51 'Berechnung %-Satz
52 GOSUB 19:GOSUB 25:GOSUB 26
53 P1=.02:PRINT:PRINT"BITTE WARTEN!
"
54 P=P1:P1=(1-(1+P)^-N)*AN/K:SOUND
15,1
55 IF ABS(P1-P)>1E-6 THEN 54 ELSE 5
6
56 GOSUB 20:GOSUB 22:GOSUB 23:GOSUB
 21:LPRINT U$:GOTO 13
57 CLS:SOUND 8,2:SOUND 6,2:SOUND 10
,2:END
```

Trigonometrische Funktionen

Programmiersprache: EBASIC. Gerätekonfiguration: HX 20, Grundversion. Speicherplatzbedarf: 675 Bytes. Im EBASIC des HX-20 ist neben SIN, COS und TAN nur die ARCUSTANGENS-FUNKTION (ATN) vorhanden. Wie löse ich den ARC SIN, ARC COS usw.? Hier die Antwort:
Die Umkehrfunktionen zu den Winkel-Funktionen kann man leicht selbst auf Grund bekannter Zusammenhänge mit Hilfe der selbst definierten Funktionen erstellen. So sind:
ARC SIN(X) = ATN(X/SQR(1-X*X)
ARC COS(X) = PI/2-ATN(x)/SQR(1-x*x)
Dabei muß man daran denken, daß einige Argumente wie 0, 90, 180 Grad zum Absturz führen, weil durch Null dividiert werden sollte. Das kann man leicht umgehen, indem man einen ganz kleinen Wert, wie 0.000000001 addiert. Das Resultat wird dadurch nicht beeinträchtigt, jedoch wird nicht mehr durch Null dividiert. Ein Absturz des Systems kann dadurch vermieden werden.
PRAKTISCHES BEISPIEL
Wir beabsichtigen, eine 5 m hohe Fernseh-Antenne aufzustellen und beschliessen,die Halteseile in einen Winkel von 35 Grad zum Flachdach anzubringen. Wenn wir drei Halteseile einsetzen, wieviel Meter Halteseil benötigen wir?



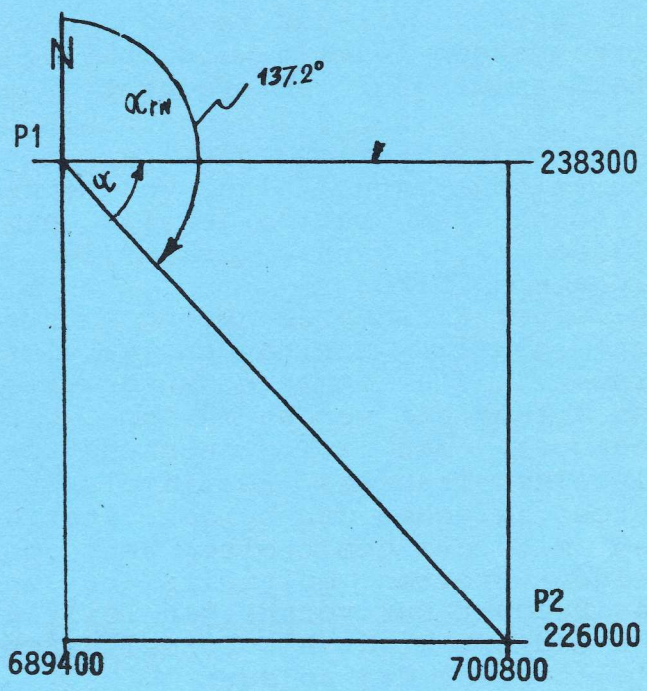
Gegenkathete
Hypothenuse = SIN(a) 5/c = SIN(35)
c = 5/(SIN(35))
Wir tippen im Direktmodus ein, nachdem wir das vorhergehende Programm laufen ließen:
? 5/FN SN(35) und erhalten
8.71724 x 3 = 26.25m
UND NOCH EINE ÜBUNG
Wir kennen die Koordinaten zweier Punkte (P1,P2) auf einer Landkarte und möchten den genauen Winkel der Richtung von P1 zu P2 kennen.Es sind die Koordinaten:
P1=689400/238300
P2=700800/226000
Wie lautet der rechtweisende Winkel (bezogen auf geografisch Nord)?Es gilt:

$$\text{Tan}(\ ) = \frac{(Y2-Y1)}{(X2-X1)}$$

$$\text{TAN}(A) = \frac{-12300}{11400} = 1.079$$

A=FNU(FN ATA(A)) = 312.825

Um den rechtweisenden Winkel zu bekommen, muß der Winkel A von 90 bzw. 450 Grad abgezogen werden. Die Funktion "FNU", die wir in Zeile 290 beifügten, reduziert die überzähligen Vollkreise, so daß kein Winkel größer als 360 Grad wird.
A = 450-A = 137.175



Pierre Weber

```
1  'Trigonometr.Funktionen
2  'für Chip-Special
3  'von Pierre Weber
15 PI=3.14159265
20 DEF FN SN(X)=SIN(X*PI/180) 'SIN
aus Winkel in Grad
25 DEF FN CS(X)=COS(X*PI/180) 'COS
aus Winkel in Grad
30 DEF FN TN(X)=TAN(X*PI/180) 'TAN
aus Winkel in Grad
35 'Umkehrfunktion mit Ausgabe in R
adian
40 DEF FN AS(X)=ATN(X/(1E-9+SQR(1-X
*X)))          'ARC SIN in Radian
45 DEF FN AC(X)=PI/2-(ATN(X/(1E-9+S
QR(1-X*X)))) 'ARC COS in Radian
50 'Umkehrfunktion mit Ausgabe in G
rad
55 DEF FN ASI(X)=FN AS(X)*180/PI 'A
RC SIN in Grad
60 DEF FN ACO(X)=FN AC(X)*180/PI 'A
RC COS in Grad
65 DEF FN ATA(X)=ATN(X)*180/PI   'A
RC TAN in Grad
70 'Reduktion auf Vollkreis
75 DEF FNU(X)=X+360-INT((X+360)/360
)*360
```

Volumen berechnen

Programmiersprache: EBASIC. Gerätekonfiguration: HX-20, Grundversion. Speicherbedarf:938 Bytes für Programm, etwa 1000 Bytes inkl. Variablen.
Dieses Programm vereinfacht eine der vielen Routineaufgaben im Bauwesen. Es dient zum Berechnen von Flächen und Volumen bezogen auf ein stationiertes Aufmaß. Das Programm arbeitet im Dialog mit dem Anwender.
DAS PROBLEM
Ein großer Teil der in der Praxis vorkommenden Aufmaße wird über ein Stationsaufmaß abgerechnet. Dies gilt für eine Flächenberechnung, wenn zur Station jeweils eine Breite oder für eine Volumenberechnung zur Station jeweils eine Fläche (z.B. aus Querprofilen) eingegeben wird.
Bei einer Flächenberechnung entstehen jeweils aneinander gereihte Trapeze, die einzeln ausgewertet und aufaddiert werden müssen. Dies bedeutet, daß zur Berechnung des einzelnen Trapezes jeweils die Differenz zwischen den beiden Stationen mit dem gemittelten Wert der beiden Breiten multipliziert und das Produkt zur vorherigen Teilfläche addiert wird. Als Formel dargestellt sieht das so aus:
S = Station
B = Breite
n = letzte Station

Fges = (S2-S1)x(B1+B2) /2 + (S3-S2)x(B2+B3)/2+...
Hier wird deutlich, daß für jede neue Teilfläche auf die bereits zuvor eingegebenen Werte zurückgegriffen werden muß. Diese Werte sind dann zweimal einzugeben. Außerdem sind zum Schluß noch einmal alle Teilflächen einzugeben, um die Gesamtsumme zu erhalten.
Bei einer Volumenberechnung gilt:
S = Station
F = Fläche
n = letzte Station
Vges = (S2-S1)x(F1+F2)/2 + (S3-S2)x(F2+F3) /2 + (Sn-S (n-1)) x (F (n-1) + Fn) /2
PROGRAMMBESCHREIUNG
Das Programm wird nach dem Eingeben mit RUN gestartet.Es fragt zuerst, ob es sich um ein Aufmaß von Station mit der dazugehörigen Breite (Flächenberechnung) oder um ein Aufmaß von Station mit der dazugehörigen Fläche (Volumenberechnung) handelt. Das Programm verzweigt je nach Entscheidung nach Adresse 110 für eine Flächenberechnung oder nach Adresse 410 für eine Volumenbereechnung und fordert zur Eingabe der erforderlichen Daten auf. Alle Werte sind nur einmal, und zwar nach aufsteigender Stationierung einzugeben.
Ab Adresse 170 beginnt die Berechnung. Die Summe der Teilflächen (EN) wird auf Null gesetzt. Über

den Inputbefehl werden die Werte eingegeben und gleich als Kontrolle ausgedruckt. In Adresse 230 wird die Teilfläche errechnet. Die Station mit Beiwert (Breite/ Fläche) wird für die nächste Berechnung neu definiert (240 + 250). Die berechnete Teilfläche wird zur bisherigen Summe addiert (260). Der GOTO- Befehl zur Adresse 200 wird erst durch die Eingabe 0,0 unterbrochen. Der Ausdruck erfolgt dann Zeile 290 für die Flächenberechnung und ab Zeile 480 für die Volumenberechnung. Das Programm ist dann beendet.

DATENEINGABE

Nach dem Einschalten erfolgt die Anzeige:

STATIONSAUFMAß
Station-Breite : B
Station-Fläche : F

Wird statt B oder F ein anderes Zeichen eingegeben, erfolgt die gleiche Anzeige erneut. Im Falle einer gewünschten Flächenberechnung erscheint:

FLäCHENBERECHNUNG
Station - Breite
?

hinter dem Fragezeichen sind die Wertepaare jeweils durch Komma getrennt einzugeben. Dezimalwerte erhalten einen Punkt, kein Komma. Wird dies nicht beachtet, erscheint "Redo" und die ganze Zeile ist neu einzugeben. Das Programm wird nicht unterbrochen. Sind alle Werte eingegeben, so ist als letztes Wertepaar einzugeben: 0,0

Das Ergebnis erscheint in der Anzeige und wird anschließend auf dem Minidrucker ausgedruckt.Im Prinzip gelten für die Volumenberechnung die gleichen Anweisungen. Für eine neue Berechnung mit RUN starten.

EINGABEFEHLER

Solange eine Eingabe nicht mit RETURN abgeschlossen ist, besteht die Möglichkeit zur Korrektur in dieser Zeile:
Löschen,Einsetzen, Ersetzen.
Ist ein Wertepaar mit RETURN abgeschlossen, kommt es bei falscher Eingabe von Dezimalpunkt oder Komma zu der Fehlermeldung Redo. Ein Fragezeichen fordert zur erneuten Eingabe der letzten Zeile auf, die ebenfalls mit RETURN abgeschlossn werden muß. Das Programm wird hierbei nicht unterbrochen. Die fehlerhafte Zeile wird ignoriert.
Zur Kontrolle wird gleich nach der Dateneingabe das Wertepaar formatiert ausgedruckt mit 3 Stellen hinter dem Dezimalpunkt: ganz gleich, ob Dezimalpunkte eingegeben wurden oder nicht. Eine falsche Ziffernfolge oder Zifferneingabe kann nach betätigter RETURN-Taste jedoch nicht mehr korrigiert werden. Ein neuer Programmstart mit RUN (bzw. PF 5) ist erforderlich. Der RUN-Befehl löscht alle vorhandenen Variablen innerhalb des Programms.

GRENZEN

Die Menge der einzugebenden Wertepaare ist unbegrenzt,da die eingegebenen Werte sofort verarbeitet werden und somit der Aufbau eines Dimensionsfeldes (DIM-Anweisung) entfällt. Die Datenausgabe ist formatiert auf 7 Stellen vor und 3 Stellen hinter dem Dezimalpunkt.

PRAXIS-BEISPIEL

Programm mit RUN starten. Gewünschte Berechnung wählen: B oder F. Station mit der jeweils dazugehörigen Breite eingeben. Die beiden Werte sind durch ein Komma voneinander zu trennen.
Abschließen der Zeile mit RETURN. Zum Beenden der Eingabe als letzte Zeile 0,0 eingeben und mit RETURN beenden. Diese Zeile wird nicht ausgedruckt.

Arno W. Wagner

```
10 'STATION -- Flächen- und Volumen
berrechnung
20 'für Chip-Special
30 'von Arno W. Wagner
40 CLS
50 B$="######.###"
60 PRINT"STATIONSAUFMASS"
70 PRINT"Station-Breite  : B"
80 PRINT"Station-Fläche  : F"
90 A$=INKEY$
100 IFA$="B"THEN130
110 IFA$="F"THEN430
120 GOTO90
130 PRINT
140 PRINT"FLACHENBERECHNUNG"
150 PRINT"Station - Breite"
```

```
160 LPRINT"    FLÄCHENBERECHNUNG"
170 LPRINT
180 LPRINT"    Station   - Breite"
190 EN=0
200 INPUTS,B
210 LPRINT USINGB$;S,B
220 INPUTSN,BN
230 IFSN=0ANDBN=0THEN300
240 LPRINT USINGB$;SN;BN
250 F=(SN-S)*(B+BN)/2
260 S=SN
270 B=BN
280 EN=EN+F
290 GOTO220
300 IFA$="F"THEN500
310 PRINT"--------------------"
320 PRINT"F in m2 :";
330 PRINT USING"#######.###";EN
340 PRINT"--------------------"
350 LPRINT
360 LPRINT"F in m2 :";
370 LPRINT USING"#######.###";EN
380 LPRINT"--------------------"
390 LPRINT
400 LPRINT
410 LPRINT
420 LPRINT:END
430 PRINT
440 PRINT"VOLUMENBERECHNUNG"
450 LPRINT"    VOLUMENBERECHNUNG"
460 LPRINT
470 PRINT"Station - Fläche"
480 LPRINT"    Station   - Fläche"
490 GOTO190
500 PRINT"--------------------"
510 PRINT"V in m3 :";
520 PRINT USING"#######.###";EN
530 PRINT"--------------------"
540 LPRINT
550 LPRINT"V in m3 :";
560 GOTO370
```

Flächen berechnen

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 890 Bytes für Programm, 1312 Bytes inkl. Variablen. Dieses Programm zur Berechnung von Flächen nach dem "Gauß-Elling"-Verfahren, setzt keine Programmierkenntnisse voraus. Es arbeitet im Dialog mit dem Anwender. Die Handhabung wird dadurch recht einfach und erfordert nach einiger Übung kein Nachschlagen in der Programmbeschreibung.

DAS PROBLEM

Die Erdmassenberechnungen im Straßen- und Kanalbau werden in der Regel über Querprofile abgerechnet. Es entstehen Vielecke, deren Eckpunkte durch die X- und Y-Koordinate bestimmt werden. Die einfachste Auswertung dieser Querprofile läßt sich mit dem "Gauß-Elling"-Verfahren abwickeln. Mit folgender Formel wird gerechnet:

```
F=0,5* ((X(1)*Y(2)-Y(1)*X(2))
       +(X(2)*Y(3)-Y(2)*X(3))
       +...
       +(X(n)*Y(1)-Y(n)*X(1)))
```

X = Punktabstand von der Achse
Y = NN-Höhe
n = letzter Punkt

Hier wird deutlich, daß bei einer manuellen Berechnung alle Koordinatenwerte zweimal eingegeben werden müssen. Außerdem sind am Schluß alle Teilergebnisse aufzuaddieren und mit 0,5 zu multiplizieren, um das Ergebnis zu bekommen.

PROGRAMMBESCHREIBUNG

Das Programm wird mit RUN gestartet und setzt mit der DIM-Anweisung in Adresse 30 und 40 je 50 Punkte für den X- und Y-Wert frei. Eine IF-Schleife in Adresse 7o überprüft, ob die eingegebene Punkteanzahl kleiner oder gleich 50 und größer als 3 ist. Sind die Kriterien erfüllt, kann mit der Koordinateneingabe begonnen werden. In Adresse 100 wird der Schleifenzähler C auf 1 gesetzt. In Adresse 110 wird der Schleifenzähler mit dem Wert der Punkteanzahl verglichen. Sind alle Punkte eingegegben, wird die Koordinateneingabe automatisch abgebrochen und die Berechnung beginnt. Mit dem Schleifenzähler C werden in Adresse 130 indizierte Variablen erzeugt. Für die Berechnung in

Adresse 170 werden diese Variablen (Koordinatenwerte) herangezogen. Die Indizierung erfolgt hierfür durch die FOR-NEXT-Schleife der Adressen 160 bis 190. Da die Koordinatenwerte des letzten Punktes mit den Werten des ersten Punktes verrechnet werden müssen,ist diese Schleife auf (N-1) begrenzt. Die letzte Rechnung erfolgt in Adresse 200. Die Aufsummierung der Teilergebnisse wird in Adresse 180 und 220 durchgeführt. Auf der Anzeige erfolgt die formatierte Ausgabe der berechneten Fläche. Der Ausdruck über den Minidrucker erfolgt auf Wunsch. Die indizierten Variablen werden in der FOR-Next-Schleife (Adresse 370 bis 390) aufgerufen und formatiert ausgedruckt.

DATENEINGABE

Nach dem Einschalten erscheint die Anzeige: Anzahl der Punkte
<=50! ?

Um eine Fläche berechnen zu können müssen mindesten drei Punkte eingegeben werden. Die maximale Punktezahl ist nur vom Programm her begrenzt. Nach dem Drücken der RETURN-Taste erscheint die Anzeige:

KOORDINATENEINGABE
X,Y
Punkt 1?

Hinter dem Fragezeichen sind die Wertepaare jeweils durch Komma getrennt einzugeben. Dezimalwerte erhalten einen Punkt, kein Komma. Wird dies nicht beachtet, erscheint 'Redo' und das Wertepaar ist neu einzugeben. Das Programm wird nicht unterbrochen. Sind alle Werte eingegeben, erfolgt automatisch die Berechnung und das Ergebnis auf dem LCD-Display.

Beispiel: F in m2: 69.969

DATEN DRUCKEN? J/N

Anschließend werden alle Daten übersichtlich formatiert auf Wunsch ausgedruckt. Für eine neue Berechnung mit RUN starten.

EINGABEFEHLER

Solange eine Eingabe noch nicht mit RETURN abgeschlossen ist, besteht die Möglichkeit zur Korrektur in dieser Zeile:
Löschen,Einsetzen,Ersetzen.
Ist ein Wertepaar mit RETURN abgeschlossen, kommt es bei falscher Eingabe von Dezimalpunkt oder Komma zu der Fehlermeldung Redo.Ein Fragezeichen fordert zur erneuten Eingabe des zuletzt eingegebenen Wertepaares auf, die ebenfalls mit RETURN abgeschlossen werden muß. Das Programm wird hierbei nicht unterbrochen. Die fehlerhafte Zeile wird ignoriert.
Eine falsche Ziffernfolge oder Zifferneingabe kann nach Betätigen der RETURN-Taste jedoch nicht mehr korrigiert werden. Ein neuer Programmstart mit RUN (bzw.PF 5) ist erforderlich. Der RUN-Befehl löscht alle vorhandenen Variablen innerhalb des Programmes.

PROGRAMMGRENZEN

Die maximale Punkteanzahl eines Querprofiles ist auf 50 begrenzt. Für umfangreiche Berechnungen kann die DIM-Anweisung entsprechend angepaßt werden. Dadurch wird aber in jedem Fall mehr Speicherplatz benötigt, auch wenn tatsächlich weniger Werte eingegeben werden. Die Berechnung selbst erfolgt mit einfacher Genauigkeit.

PRAXIS-BEISPIEL

Programm mit RUN starten. Anzahl der Punkte eingeben. RETURN-Taste drücken. Koordinatenwerte eingeben X- und Y-Wert durch Komma trennen. RETURN-Taste drücken. Nach der Koordinateneingabe des letzten Punktes erscheint auf der LCD-Anzeige das Ergebnis. Wird J gedrückt, gibt der Minidrucker alle Daten auf dem Kontrollstreifen aus.

Arno W. Wagner

Beispiel

```
-----------------------
Flächenberechnung nach
Gauß - Ellipse

Punkt-Nr.    X        Y
        1  -7.500  184.500
        2  -5.500  184.500
        3  13.500  185.000
        4  16.600  185.000
        5  12.400  180.900
        6  12.100  180.050
        7   2.600  180.050
        8   0.550  183.470
        9  -7.050  183.470

Fläche in m2:     69.969
-----------------------
```

```
10 'ELLING -- Flächenberechnung
20 'für Chip-Special
30 'von Arno W. Wagner
40 CLS
50 DIMX(50)
60 DIMY(50)
70 PRINT"ANZAHL DER PUNKTE"
80 INPUT"<=50!          ";N
90 IFN<3ORN>50THEN70
100 PRINT
110 PRINT"KOORDINATENEINGABE
                              X,Y"
120 C=1
130 IFC=(N+1)THEN180
140 PRINT"PUNKT";C;
150 INPUT X(C),Y(C)
160 C=C+1
170 GOTO 130
180 FOR Z=1 TO N-1
190 SU=X(Z)*Y(Z+1)-Y(Z)*X(Z+1)
200 SX=SX+SU
210 NEXTZ
220 SE=X(N)*Y(1)-Y(N)*X(1)
230 PRINT"--------------------"
240 F=(SX+SE)/2
250 PRINT"F in m2 :"::PRINT USING"#
######.###";ABS(F)
260 PRINT"--------------------"
270 PRINT"DATEN DRUCKEN ? J/N"
280 A$=INKEY$
290 IFA$="J"THEN320
300 IFA$="N"THENEND
310 GOTO280
320 B$="####.###"
330 C$="########"
340 LPRINT"------------------------
"
350 LPRINT"Flächenberechnung nach"
360 LPRINT"Gauß - Elling"
370 LPRINT
380 LPRINT"Punkt-Nr.    X         Y"
390 FOR I=1 TO N
400 LPRINT USING C$;I;: LPRINT USIN
G B$;X(I);Y(I)
410 NEXTI
420 LPRINT
430 LPRINT"Fläche in m2:"::LPRINT U
SING"#######.###";ABS(F)
440 LPRINT"------------------------
"
450 LPRINT
460 LPRINT
470 LPRINT
480 LPRINT
```

Prismen

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion, Epson-Matrixdrucker FX-80, RS 232-Interface 8145, Anschlußkabel # 714. Speicherplatzbedarf:4161 Bytes für Programm, 11848 Bytes inkl. Variablen.
Aufgabe dieses Programmes ist es, eine Massenberechnung aus Prismen mit Hilfe des Hand-Held-Computers HX-20 durchzuführen. Die Verfahrensbeschreibung 22.013 der Richtlinien für die Elektronische Bauabrechnung (REB-VB 22.013),herausgegeben von der Forschungsgesellschaft für das Straßenwesen, soll hier als Richtlinie gelten. Dies gilt besonders für die Ausgabe der Kontrollwerte sowie für die errechneten Werte. Daß ein kleiner tragbarer Computer nicht allen Anforderungen dieser Verfahrensbeschreibung nachkommen kann, versteht sich von selbst. Deshalb wurde bei Erarbeitung dieses Programmes nur das Wesentliche und für die Praxis Notwendige berücksichtigt:

* Schnelle und übersichtliche Dateneingabe über Tastatur oder von der Mikro-Kassette.
* Sofortige Berechnung der Werte und Ausgabe über die LCD-Anzeige.
* Die verwendeten Formeln und Rechenansätze entsprechen der REB VB 23.003
* Ausdruck der Koordinaten, Dreiecksmaschen, Teilgrundfläche, Teilvolumina und der Gesamtergebnisse entsprechend der REB VB 23.003 (Auf die Berechnung der Oberfläche wurde verzichtet).
* Sofortige Sicherung der eingegebenen Daten durch Abspeicherung auf Band.
* Nachträgliche Korrekturmöglichkeiten bereits abgespeicherter Daten werden beschrie-

ben.
Dieses Programm zur Berechnung von Massen nach der Prismenmethode setzt keine Programmierkenntnisse voraus. Es arbeitet im Dialog mit dem Anwender. Die Handhabung wird dadurch übersichtlich und erfordert nach einiger Übung kein Nachschlagen in der Programmbeschreibung.
DAS PROBLEM
Meistens wird zur Erdmassenberechnung das Gauß-Elling-Verfahren mit stationierten Querprofilen angewendet. Von Vorteil ist dieses Verfahren zweifellos für langgezogene Dämme oder Einschnitte. Selbst in Kurvenbereichen lassen sich über den Schwerpunktabstand brauchbare Ergebnisse erzielen. Sind die zu berechnenden Körper aber kubischer Natur, dann wird das Verfahren mit stationierten Querprofilen zu aufwendig. Dies gilt hauptsächlich für Gewinnungsgebiete,Staubecken und Halden. Grundlage für die Massenberechnung aus Prismen ist ein Lageplan, in dem alle markanten Geländepunkte als dreidimensionale Koordinaten (X, Y, Z) bekannt sind. Die Punkte werden so durch Linien miteinander verbunden, daß nur Dreiecke entstehen (die Dreiecksmaschen). Greift man jetzt willkürlich ein Dreieck heraus und verbindet die drei Eckpunkte mit einer waagerechten Bezugsebene, dann erhält man ein Dreieckprisma mit einer rechtwinklig abschließenden Grundfläche und einer schiefen ebenen Deckfläche(vgl. Bild 1,Seite 44):
Das Volumen dieses Dreieckprismas läßt sich leicht bestimmen:
Grundfläche * mittlere Höhe von G bis FU
Nimmt man an, daß das Gelände nach Beendigung der Bauarbeiten auf diesem speziellen Dreiecksfleck die in Bild 2 gezeigte Lage (FN) hat, läßt sich das Volumen zwischen Deckfläche FU und Deckfläche FN aus der Differenz beider Prismen bestimmen:
Grundfläche * mittlere Höhe von G bis FU
Grundfläche * mittlere Höhe von G bis FN
Die Bezugsebene muß von allen Teildeckflächen weit genug entfernt sein. Wenn man sich vorstellt, daß die Deckflächen meist aus einigen Hundert Dreiecken bestehen, erkennt man den großen Rechenaufwand der dahinter steckt. Abgesehen davon, müssen bei manueller Berechnung alle Dreieckspunkte für jedes Dreieck komplett eingegeben werden, obwohl für die anliegenden Dreiecke ein bis zwei Punkte wieder verwendet werden könnten.
Um gleich bei der Eingabe die Arbeit zu vereinfachen, werden die Dreiecke paarweise behandelt. Die Dreieckskoordinaten werden alle einmal nach aufsteigender Numerierung eingegeben und für die weitere Berechnung nach Anweisung aufgrund der Dreiecksmascheneingabe als indizierte Variablen vom Programm in die Berechnung gerufen.
Folgende Formeln werden für die Berechnung verwendet:
Grundfläche Dreieckprisma:

```
G = ABS (1/2(X1*(Y2-Y3)+ X2* (Y3-Y1)
+X3*(Y1-Y2)))
```

Volumen Dreieckprisma:

```
V = 1/3* (Z1+Z2+Z3-3*HO)*G
```

X,Y,Z: dreidimensionale Koordinaten
1,2,3,... : Punktnummern
HO : Höhe der Bezugsebene
PROGRAMMBESCHREIBUNG
Das Programm wird mit RUN gestartet und setzt mit den DIM-Anweisungen in Adresse 50 Platz für 300 dreidimensionale Koordinaten und 200 Dreiecksmaschen. In Adresse 60 werden Formate für die PRINT-USING-Befehle der tabellarischen Aufstellung festgelegt. Der Computer fragt in Adresse 110 über den INKEY$-Befehl die Art der Dateneingabe ab. Sollen die Daten per Band eingespielt werden, springt das Programm in die Adresse 1360. Mit dem INPUT-Befehl in Adresse 1390 werden die Variablen H$, N, Z und HO eingelesen. H$ = String für die Überschrift. N = Anzahl der Punkte für die Koordinateneingabe, Z = Anzahl der Dreiecksmaschen, HO = Bezugshorizont. Die folgende FOR-NEXT-Schleife (1400-1420) liest die Koordinaten bis zur N-ten Punkt-Nr. Danach werden die Dreiecksmaschen ebenfalls mittels einer FOR-NEXT-Schleife (1430 bis 1450) bis zur Z-ten Masche eingelesen. Das Programm springt zurück bis zur Adresse 390.

Die Berechnung beginnt.
Sollen die Daten per Tastatur eingegeben werden, springt das Programm in die Adresse 150. Die gleichen Daten, wie bei der Bandeinspielung beschrieben, sind einzutippen. Die Abfrage für Koordinateneingabe bzw. Dreiecksmascheneingabe wird durch eine GOTO-Anweisung ständig mit aufsteigender Numerierung so lange wiederholt, bis als erster Wert eine 0 eingegeben wird. Die IF-Abfragen in Adresse 230 bzw. Adresse 360 brechen die GOTO-Schleife in diesem Falle ab.
Sind alle Werte eingegeben, springt das Programm von Adresse 360 zur Adresse 380. Die Berechnung beginnt. In Adresse 510 wird jeweils die erste Fläche eines Dreieckpaares berechnet. In Adresse 520 werden die Dreieckteilflächen aufaddiert. Das Volumen wird in Adresse 530 und die mittlere Höhe in Adresse 540 berechnet. Die Ergebnisse werden in Adresse 550 formatiert über die LCD-Anzeige ausgegeben. Die Aufaddition der Teilvolumina erfolgt in Adresse 560. Für den Fall, daß kein Dreieckpaar eingegeben wurde, sondern nur ein Dreieck, springt das Programm ab Adresse 570 zur Adresse 630.
Die Berechnung des zweiten Prismas eines jeden Dreieckpaares beginnt in Adresse 580 und endet in Adresse 620. Hier werden die beiden letzten Punkte des ersten Dreiecks: 2 und 3 wieder verwendet. Als neuer Punkt fließt nur Nr.4 ein. Ist der erste Wert einer Dreiecksmascheneingabe =0, wird die FOR-NEXT-Schleife (500-630) abgebrochen. Das Gesamtergebnis wird über die LCD-Anzeige ausgegeben. Sollen die Daten ausgedruckt werden, spingt das Programm in Adresse 730. Der Ausdruck über den Matrixdrucker beginnt. Die erforderlichen Ergebnisse werden hierzu wieder neu berechnet, da zusätzliches Reservieren der bereits zuvor errechneten Werte zuviel Speicherplatz in Anspruch nehmen würde. Die Druckroutine endet in Adresse 1150. Die Routine zur Datenabspeicherung über den internen Mikro-Kassettenrecorder (1260 bis 1350) gleicht der Routine zur Dateneinspielung, die anfangs beschrieben wurde.

DRUCKER-ANPASSUNG

Die Verbindung der beiden Geräte erfolgt durch das Anschlußkabel #714. Beim Anschluß des Kabels müssen die Geräte ausgeschaltet sein. Im FX-80 muß das RS-232-Interface 8145 installiert sein. Mit dem OPEN-Befehl in Adresse 730 wird die Schnittstelle RS 232 angesprochen (COM0) und die Datenübertragungsrate spezifiziert (48N22). Der Datenübertragungsschlüssel lautet:

```
4 = Baud Rate      : 1200 bps
8 = Wortlänge      : 8 Bit Wort
N = Parity         : No Parity Check
2 = Stop Bits      : 2 Stop Bits
2 = Aktive         : CTS = aktiv
    Kontroll-        DSR = aktiv
    Leitung          RTS = + aktiv
                     CD  = aktiv
```

Dieser Übertragungsschlüssel weicht von der standardmäßig eingestellten Version ab. Damit die übertragenen Daten auch vom Matrixdrucker FX-80 empfangen werden können, müssen die DIP-Schalter mit dem Datenübertragungsschlüssel aus dem OPEN- Befehl abgestimmt werden. Siehe hierzu FX-80-Handbuch Seite 40-46. Folgende DIP-Schalterstellungen sind herzustellen:

```
Grundplatine FX-80 DIP 1:01010101
                   DIP 2:1110
8145-Platine       DIP 1:10101111
(RS232)            DIP 2:0101
1 = ON   ;   0 = OFF
```

DATENEINGABE

Nach dem Einschalten mit RUN bzw. PF5 starten. Folgende Anzeige erscheint:

```
MASSENBERECHNUNG AUS PRISMEN
        DATENEINGABE ÜBER
        TASTATUR/BAND  T/B
```

Soll die Eingabe über die Tastatur erfolgen, erscheint nach dem Drükken der T-Taste die Aufforderung zur Eingabe der Überschrift bzw. des Projektnamens:

```
ÜBERSCHRIFT/PROJEKT
?
```

Ist die Überschrift eingegeben, wird mit der RETURN-Taste bestätigt. Auf der LCD-Anzeige erscheint folgender Text:

```
      KOORDINATENEINGABE
                Y, X. Z
      PUNKT  1  ?
```

Die dreidimensionalen Geländepunkte sind einzugeben. Die Punktnummerierung erfolgt automatisch aufsteigend, nachdem jeweils die Eingabe für einen Punkt durch die RETURN-Taste bestätigt wurde. Die Werte für Y, X und Z sind jeweils durch ein Komma zu trennen. Dezimalwerte erhalten einen Punkt, kein Komma. Wird dies nicht beachtet, erscheint 'Redo' und die ganze Zeile ist neu einzugeben. Das Programm wird dabei nicht unterbrochen. Sind alle Punkte eingegegben, so ist als letzte Zeile: 0,0,0 einzugeben. Andere Punktnummerierungen als die soeben vom Programm her festgelegten, können nicht berücksichtigt werden. Nach beendeter Koordinateneingabe ist der Bezugshorizont einzugeben. Folgender Ausdruck erscheint auf der LCD-Anzeige:

```
ENDE KOORDINATENEINGABE
----------------------
KONSTANTER BEZUGSHORIZONT
?
```

Die Eingabe ist mit RETURN zu bestätigen. Das Programm fordert zur Eingabe der Dreiecksmaschen auf:

```
DREIECKSMASCHENEINGABE
     P1, P2, P3, P4
DREIECKPAAR 1 ?
```

Anders als bei der Koordinateneingabe, muß die Eingabe der Dreiecksmaschen nach genauen Regeln erfolgen (Bild 3,Seite 45).

```
  Punktverbindung Dreieck 1A: 1-2-7
+ Punktverbindung Dreieck 1B: 2-7-6
-----------------------------------
= Punktverbindung
  Dreieckpaar 1:              1-2-7-6
```

Auf der LCD-Anzeige muß dann folgende Zeile stehen:

```
DREIECKPAAR  1 ? 1, 2, 7, 6
```

Die Zeile ist mit RETURN zu bestätigen.
Die weiteren Zeilen lauten demnach:

```
Dreieckpaar   2 ? 2, 5, 6, 4
Dreieckpaar   3 ? 2, 3, 5, 4
Dreieckpaar   4 ? 3, 4, 8, 0
```

Da es kein viertes Dreieckpaar gibt, ist für den Wert P4 eine 0 einzugeben. Um die Dreiecksmascheneingabe zu beenden, ist die folgende Zeile einzugeben: Dreieckpaar 5 ? 0, 0, 0, 0
Sobald die Zeile mit RETURN bestätigt wird, beginnt die Berechnung. Auf der LCD-Anzeige erscheinen Zeile für Zeile die errechneten Werte. Mit der Taste PAUSE kann das Programm angehalten, und durch Drücken einer beliebigen Taste (außer BREAK) fotgeführt werden. Unterbricht man den Berechnungsablauf mit der BREAK-Taste, kann der Cursor nach rechts bewegt werden und die rechte Tabellenhälfte wird sichtbar. Zur Fortführung des Programmes ist der Cursor in die unterste Zeile zu bringen und anschließend der Befehl CONT<RETURN> einzugeben. Das Programm fährt fort. Sind alle Daten über die LCD-Anzeige gelaufen, fragt das Programm, ob die Daten ausgedruckt werden sollen:

```
DATEN DRUCKEN  ?
(J/N)
?
```

Als letzte Abfrage erscheint:

```
DATEN SPEICHERN  ?
(J/N)
```

Im Falle einer Datenspeicherung druckt der Minidrucker die Überschrift bzw. Projektname und den Zählerstand zu Beginn der Abspeicherung aus. Vor jeder Abspeicherung ist zu überprüfen, ob auch die geeignete Kassette eingelegt wurde. Daten, die bereits auf dieser Stelle gespeichert sind, werden sonst gelöscht.
Bei Eingabe über Band ist die Mikro-Kassette entsprechend zu positionieren. Nach dem Drücken der B-Taste erscheint auf der LCD-Taste folgender Text:

```
EINLESEN DER DATEN VOM BAND
```

Nach dem Einlesen wird automatisch mit der Berechnung begonnen. Der weitere Ablauf ist mit der Tastatureingabe identisch.

```
EINGABEFEHLER
```

Solange eine Eingabe noch nicht mit RETURN abgeschlossen ist, besteht die Möglichkeit zur Korrektur in dieser Zeile:

Löschen, Einsetzen, Ersetzen.

Ist eine Datenzeile mit RETURN abgeschlossen, kommt es bei falscher Eingabe von Dezimalpunkt oder

Komma zu der Fehlermeldung 'Redo'. Ein Fragezeichn fordert zur erneuten Eingabe der letzten Zeile auf, die ebenfalls mit RETURN abgeschlossen werden muß. Das Programm wird hierbei nicht unterbrochen. Die fehlerhafte Zeile wird ignoriert.
Für Eingabefehler, die man erst nach der Berechnung bzw. beim Ausdrucken festgestellt, gibt es auch eine Möglichkeit der Korrektur. Hierzu sind gezielt die Werte der betroffenen Variablen-Namen neu zu definieren. Bevor man mit dieser Korrektur beginnt, sollten die Werte erst einmal abgespeichert werden. Wird nämlich versehentlich die LIST- oder RUN-Taste gedrückt, sind alle Daten im Hauptspeicher gelöscht. Dann müssen die Daten wieder neu eingegeben werden, und das ist vom Band aus viel einfacher. Da die innerhalb des Programmes verwendeten Variablen-Namen nur selten mit denen in der Praxis bezeichneten Namen übereinstimmen, sind hier die verwendeten Bezeichnungen gegenübergestellt.

Koordinatenwert: Y (P) = Variablen-Name X (P)
Koordinatenwert: X (P) = Variablen-Name Y (P)
Koordinatenwert: Z (P) = Variablen-Name Z (P)

Für P ist die entsprechende Punkt-Nr. einzusetzen.
Dreiecksmaschen:

P (N,1) = Variablen-Name D (N,1)
P (N,2) = Variablen-Name D (N,2)
P (N,3) = Variablen-Name D (N,3)
P (N,4) = Variablen-Name D (N,4)

Für N ist die entsprechende Flächen-Nr. einzusetzen.
Im Ausdruck sind die Dreieckspaare schon jeweils getrennt aufgeführt. Der Punkt 4 entspricht daher dem Wert P3 in dem Nachbardreieck:

	P1,	P2,	P3,	P4
Dreieckpaar 1 ?	1,	2,	12,	3

= Ausdruck:

Nr.	P1	P2	P3
1	1	2	12
1	2	12	3

Soll beispielsweise der Y-Wert des zehnten Punktes (vgl. Praxis-Beispiel) in eine 55.2 umgewandelt werden, dann ist das Programm mit BREAK zu unterbrechen und X(10) =55,2 < RETURN einzutippen. Mit CONT < RETURN > kann wieder im Programm fortgefahren werden. Hierzu muß dann die letzte Abfrage beantwortet werden (Daten drucken bzw. Daten speichern).

GRENZEN

Die Menge der maximalen Geländepunkte ist für die 16-K-Version (Grundausstattung) auf 300 festgelegt. Die Anzahl der maximalen Dreiecksmaschen auf 200 pro Paar. Es wird mit einfacher Genauigkeit gerechnet. Die Datenausgabe für die errechneten Werte ist formatiert auf 8 Stellen vor und 3 Stellen hinter dem Komma. Das Programm erlaubt jeweils nur die Berechnung von einer Prismen-Serie. Für das endgültige Ergebnis, Erdabtrag oder Erdauftrag, sind deshalb zwei Berechnungen erforderlich, deren Ergebnisse voneinander zu subtrahieren sind.

PRAXIS-BEISPIEL

Für die nachfolgende Aufmaßskizze wird eine Berechnung durchgeführt. Das errechnete Volumen liegt zwischen den Flächen Urgelände und Bezugsebene.
Vorgehensweise: Programm mit RUN starten. Eingabe der Daten über Tastatur oder Band. Nach Beendigung der Dreiecksmascheneingabe erfolgt automatisch die Berechnung. Für den Ausdruck auf dem Matrixdrucker ist die J-Taste zu drücken und mit RETURN zu bestätigen. Nach dem Ausdruck lassen sich die verwendeten Eingabedaten, nicht deren Ergebnisse, abspeichern. Zuvor ist die Kassette auf die richtige Stelle zu spulen. Für weitere Berechnungen RUN RETURN bzw. PF5 eingeben. Alle bisher verwendeten Daten werden im Hauptspeicher gelöscht.

Arno W.Wagner

PX-Tips für Epson PX-8

Das Büchlein soll dem Anwender die Möglichkeit geben, seinen PX-8 - das verwandte, aber nicht kompatible Schwestermodell zum HX-20 - nach den eigenen individuellen Bedürfnissen zu nutzen. 1. Auflage 1984. 226 Seiten, DIN-A-5-Format. Herausgeber: Symic Computer GmbH, Möchengladbach. Inklusive beigepackter Demo-Kassette 89 Mark.

```
10 'PRISMEN
20 'für Chip-Special
30 'von Arno W. Wagner
40 REM SPEICHERPLATZ OHNE VARIABLEN
  4176 BYTES
50 REM SPEICHERPLATZ INCL.VARIABLEN
 11863 BYTES
60 CLS
70 DIMX(300):DIMY(300):DIMZ(300):DI
MD(200,4)
80 C$="#####.###":D$="###":F$="####
####.###":G$="#####"
90 PRINT"MASSENBERECHNUNG AUS"
100 PRINT"      PRISMEN"
110 PRINT"DATENEINGABE  ÜBER"
120 PRINT"TASTATUR/BAND  T/B":
130 B$=INKEY$
140 IFB$="B"THEN1380
150 IFB$="T"THEN170
160 GOTO130
170 CLS:PRINT"ÜBERSCHRIFT/PROJEKT"
180 LINE INPUT H$
190 CLS:PRINT"KOORDINATENEINGABE"
200 PRINT"          Y , X , Z"
210 N=0
220 N=N+1
230 PRINT"Punkt";N;
240 INPUTX(N),Y(N),Z(N)
250 IFX(N)=0THEN270
260 GOTO220
270 PRINT"ENDE KOORDINATENEINGABE"
280 PRINT"-----------------------"
290 PRINT"KONSTANTER BEZUGSHORIZONT
"
300 INPUTHO
310 PRINT"-----------------------"
320 PRINT"DREIECKSMASCHENEINGABE"
330 PRINT"      P1,P2,P3,P4"
340 Z=0
350 Z=Z+1
360 PRINT"DREIECKPAAR";Z;
370 INPUTD(Z,1),D(Z,2),D(Z,3),D(Z,4
)
380 IFD(Z,1)=0THEN400
390 GOTO350
400 PRINT"ENDE DREIECKSMASCHENEINGA
BE"
410 PRINT"-------------------------
--"
420 PRINT"BERECHNUNG LÄUFT"
430 PRINT"-------------------------
--"
440 FG=0:VK=0
450 PRINT"MASSENBERECHNUNG AUS "
460 PRINT"      PRISMEN"
470 PRINT"--------------------"
480 PRINT
490 PRINT"BEZUGSHORIZONT:";HO:PRINT
500 PRINT" NR P1 P2 P3   M.HÖHE   F
LÄCHE  VOLUMEN"
510 FORM=1TOZ
520 IFD(M,1)=0THEN660
530 FD=ABS(.5*(X(D(M,1))*(Y(D(M,2))
-Y(D(M,3)))+X(D(M,2))*(Y(D(M,3))-Y(
D(M,1)))+X(D(M,3))*(Y(D(M,1))-Y(D(M
,2)))))
540 FG=FG+FD
550 VP=(Z(D(M,1))+Z(D(M,2))+Z(D(M,3
))-3*HO)*FD/3
560 MH=VP/FD
570 PRINT USING D$;M,D(M,1),D(M,2),
D(M,3);:PRINT USING C$;MH,FD,VP
580 VK=VK+VP
590 IFD(M,4)=0THEN650
600 FD=ABS(.5*(X(D(M,2))*(Y(D(M,3))
-Y(D(M,4)))+X(D(M,3))*(Y(D(M,4))-Y(
D(M,2)))+X(D(M,4))*(Y(D(M,2))-Y(D(M
,3))))
610 FG=FG+FD
620 VP=(Z(D(M,2))+Z(D(M,3))+Z(D(M,4
))-3*HO)*FD/3:MH=VP/FD
630 PRINT USING D$:M,D(M,2),D(M,3),
D(M,4);:PRINT USING C$;MH,FD,VP
640 VK=VK+VP
650 NEXT M
660 PRINT
670 PRINT"GESAMT                  ";:P
RINT USING C$:FG,VK
680 PRINT"**********************"
690 PRINT"   DATEN DRUCKEN  ?"
700 PRINT"      ( J/N )"
710 INPUT E$
720 IFE$="J"THEN750
730 IFE$="N"THEN1180
740 GOTO690
750 OPEN"O",#2,"COM0:(48N22)"
760 WIDTH "COM0:",255
770 PRINT#2,"       ";H$
780 PRINT#2,"   ..................
.....................................
................."
790 PRINT#2,"       KOORDINATENVERZE
ICHNIS FÜR MASSEN AUS PRISMEN (Prog
r.A.W.Wagner)"
800 PRINT#2,"   ..................
.....................................
................."
810 PRINT#2
820 PRINT#2,"       PUNKT NR.
Y            X            Z"
830 FORL=1TON
840 PRINT#2,"            ";:PRINT#2,
USING D$;L;:PRINT#2,USING F$;X(L),Y
(L),Z(L)
```

```
850 NEXTL
860 PRINT#2,"    ====================
====================================
================"
870 PRINT#2,CHR$(12);
880 PRINT#2,"       ";H$
890 PRINT#2,"    ....................
....................................
.................."
900 PRINT#2,"       MASSENBERECHNUNG
 AUS PRISMEN              (Progr.A.W.
Wagner)"
910 PRINT#2,"    ....................
....................................
.................."
920 PRINT#2
930 PRINT#2,"       BEZUGSHORIZONT
:"::PRINT#2,USING C$;HO
940 PRINT#2
950 PRINT#2,"       Nr   P1   P2   P3
    m.Höhe       Fläche       Volumen"
960 PRINT#2
970 FG=0:VK=0
980 FORM=1TOZ
990 IFD(M,1)=0THEN1130
1000 FD=ABS(.5*(X(D(M,1))*(Y(D(M,2)
)-Y(D(M,3)))+X(D(M,2))*(Y(D(M,3))-Y
(D(M,1)))+X(D(M,3))*(Y(D(M,1))-Y(D(
M,2)))))
1010 FG=FG+FD
1020 VP=(Z(D(M,1))+Z(D(M,2))+Z(D(M,
3))-3*HO)*FD/3
1030 MH=VP/FD
1040 PRINT#2,"     ";:PRINT#2,USING
G$;M,D(M,1),D(M,2),D(M,3);:PRINT#2,
USING F$;MH,FD,VP
1050 VK=VK+VP
1060 IFD(M,4)=0THEN1120
1070 FD=ABS(.5*(X(D(M,2))*(Y(D(M,3)
)-Y(D(M,4)))+X(D(M,3))*(Y(D(M,4))-Y
(D(M,2)))+X(D(M,4))*(Y(D(M,2))-Y(D(
M,3)))))
1080 FG=FG+FD
1090 VP=(Z(D(M,2))+Z(D(M,3))+Z(D(M,
4))-3*HO)*FD/3:MH=VP/FD
1100 PRINT#2,"     ";:PRINT#2,USING
G$;M,D(M,2),D(M,3),D(M,4);:PRINT#2,
USING F$;MH,FD,VP
1110 VK=VK+VP
1120 NEXT M
1130 PRINT#2
1140 PRINT#2,"       GESAMT
          ";:PRINT#2,USING F$;FG,V
K
1150 PRINT#2,"    ====================
====================================
================"
1160 PRINT#2,CHR$(12);
1170 CLOSE
1180 CLS:PRINT" DATEN SPEICHERN  ?"
1190 PRINT"    ( J/N )"
1200 A$=INKEY$
1210 IFA$="J"THEN 1250
1220 IFA$="N"THENEND
1230 GOTO1200
1240 END
1250 LPRINT"PRISMEN-DATEI"
1260 LPRINT H$
1270 LPRINT "START-NR.:  ",TAPCNT:L
PRINT
1280 OPEN"O",#1,"CAS0:PDATEI"
1290 PRINT#1,H$:PRINT#1,N:PRINT#1,Z
:PRINT#1,HO
1300 FORL=1TON
1310 PRINT#1,X(L):PRINT#1,Y(L):PRIN
T#1,Z(L)
1320 NEXTL
1330 FORM=1TOZ
1340 PRINT#1,D(M,1):PRINT#1,D(M,2):
PRINT#1,D(M,3):PRINT#1,D(M,4)
1350 NEXTM
1360 CLOSE
1370 END
1380 CLS:PRINT"EINLESEN DER DATEN"
1390 PRINT"VOM BAND"
1400 OPEN"I",#1,"CAS0:PDATEI"
1410 INPUT#1,H$,N,Z,HO
1420 FORL=1TON
1430 INPUT#1,X(L),Y(L),Z(L)
1440 NEXTL
1450 FORM=1TOZ
1460 INPUT#1,D(M,1),D(M,2),D(M,3),D
(M,4)
1470 NEXTM
1480 CLOSE
1490 GOTO410
```

Bild 1

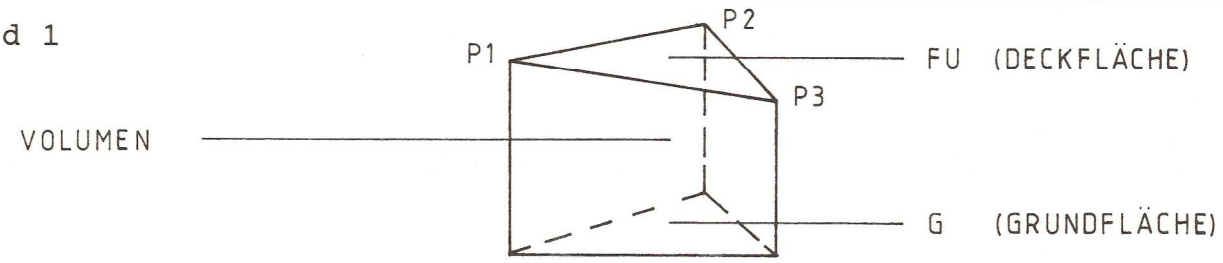


Bild 2

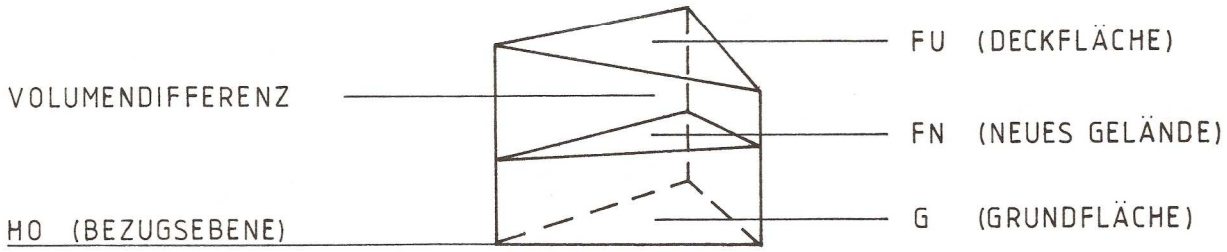
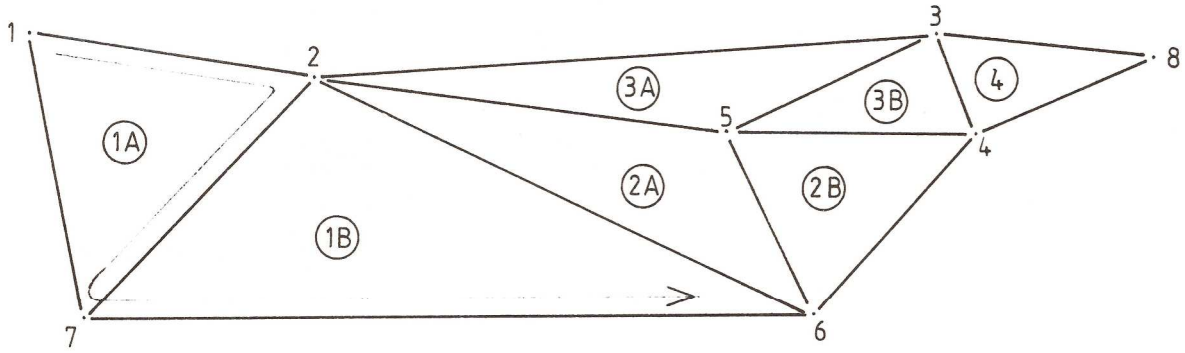


Bild 3



Beispiel

Ausdruck der Kontrollwerte und der Volumenberechnung

PROJEKT ERZGRUBE A-DORF: URGELÄNDE

KOORDINATENVERZEICHNIS FÜR MASSEN AUS PRISMEN

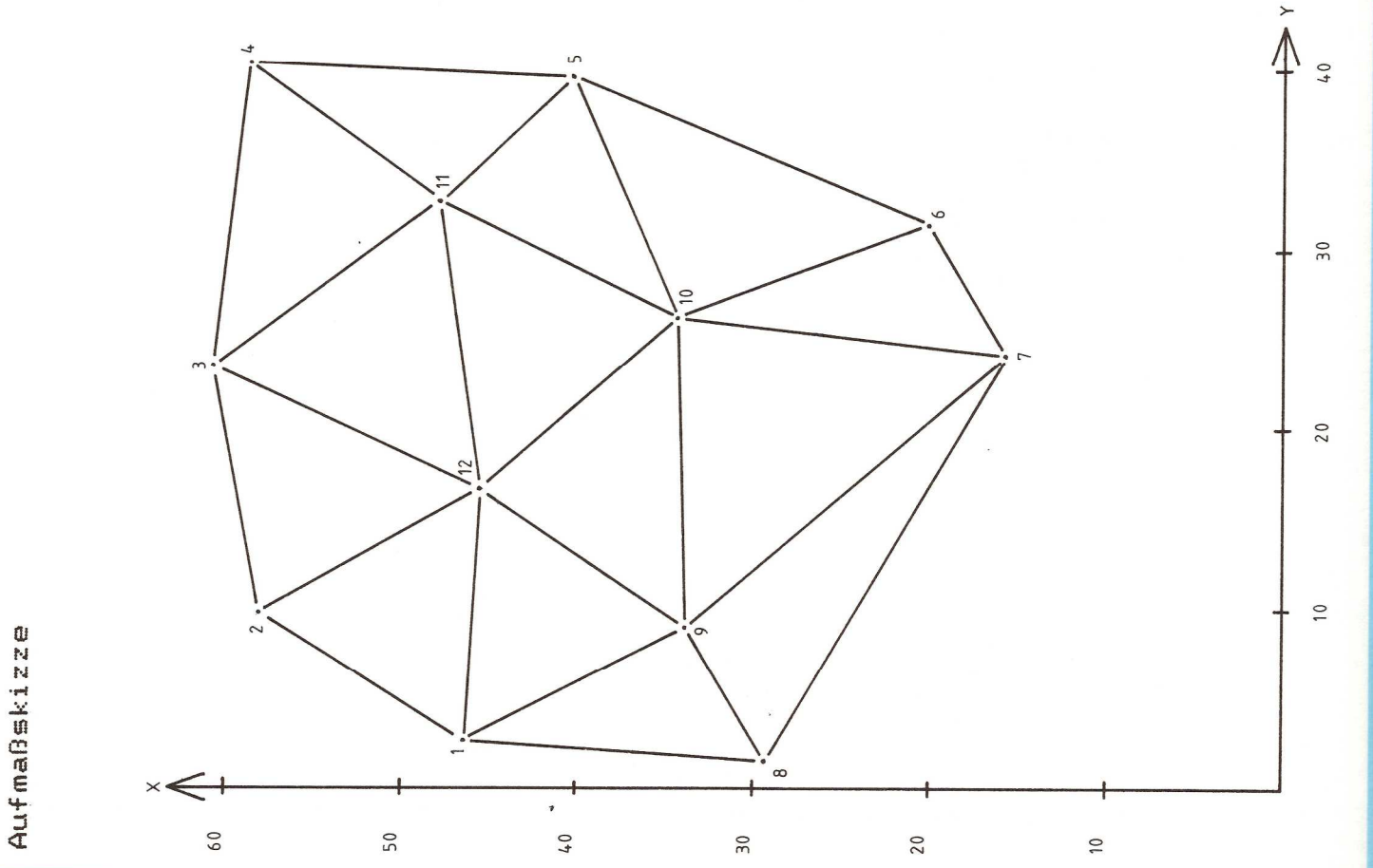
PUNKT NR.	Y	X	Z
1	2.300	46.700	9.300
2	9.400	58.600	10.200
3	23.200	61.100	9.700
4	40.300	59.200	8.500
5	39.600	40.700	7.200
6	31.400	20.200	6.000
7	24.200	15.800	6.600
8	1.200	29.500	7.700
9	8.700	34.100	7.100
10	26.200	34.600	8.000
11	32.600	48.300	8.800
12	16.400	45.800	9.300
13	0.000	0.000	0.000

PROJEKT ERZGRUBE A-DORF: URGELANDE Beispiel

MASSENBERECHNUNG AUS PRISMEN

BEZUGSHORIZONT : 3.000

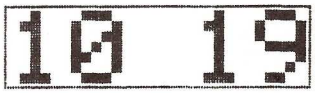
Nr	P1	P2	P3	m.Höhe	Fläche	Volumen
1	1	2	12	6.600	87.090	574.794
1	2	12	3	6.733	97.070	653.605
2	12	3	11	6.267	115.430	723.361
2	3	11	4	6.000	100.510	603.060
3	4	11	5	5.167	67.410	348.285
3	11	5	10	5.000	72.270	361.350
4	5	6	10	4.067	112.340	456.849
4	6	10	7	3.867	63.280	244.683
5	10	7	9	4.233	164.000	694.267
5	7	9	8	4.133	104.275	431.003
6	8	9	1	5.033	61.970	311.916
6	9	1	12	5.567	85.950	478.455
7	9	10	12	5.133	100.450	515.644
7	10	12	11	5.700	102.970	586.929
GESAMT					1335.020	6984.200



Digitaluhr

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 490 Bytes.
Dieses Programm stellt auf dem ganzen LCD-Display die Uhrzeit in großen, aus mehreren Metern Distanz lesbaren Ziffern dar. Die Stunden und Minuten werden hierzu in die Mitte der Anzeige mit PRINT geschrieben, dann punktweise abgetastet, um in vierfacher Größe als Graphik aufgezeichnet zu werden. Damit Zeit gespart werden kann - denn Graphik ist ja nicht sehr schnell - werden nur die Stellen der Zeit neu geschrieben, die sich seit dem vorherigen Mal geändert haben. Die Bedienung ist einfach: Eintippen, mit RUN starten.

Beat Dörr

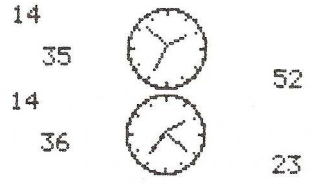


```
1 'Digitaluhr
2 'für Chip-Special
3 'von Beat Dörr
4 PRINT CHR$(23);:DEFINT H-M:DEFSTR
 T:CLS:LINE(0,0)-(119,0),PSET:LINE-
(119,31),PSET:LINE-(0,31),PSET:LINE
-(0,0),PSET
5 TI=LEFT$(TIME$,5):IF T=TI THEN 5
ELSE LOCATE 9,1:PRINT LEFT$(TI,2):L
OCATE 9,2:PRINT RIGHT$(TI,2):FOR H=
0 TO 1:FOR I=0 TO 1:IF MID$(T,H*3+I
+1,1)<>MID$(TI,H*3+I+1,1) THEN GOSU
B 7
6 NEXT:NEXT:LOCATE 9,1:PRINT"  ";:L
OCATE 9,2:PRINT"  ";:T=TI:GOTO 5
7 HP=H*8+8:IP=I*6+54:IL=H*72+I*24+2
:FOR L=IL TO IL+19:LINE(L,2)-(L,30)
,PRESET:NEXT:FOR J=0 TO 6:JP=HP+J:J
L=J*4+2:FOR K=0 TO 4:IF POINT(IP+K,
JP)=1 THEN KL=IL+K*4:FOR L=JL TO JL
+3:LINE(KL,L)-(KL+3,L),PSET:NEXT
8 NEXT:NEXT:RETURN
```

Analoguhr

Programmiersprache: EBASIC. Gerätekonfiguration HX20, Gundversion. Speicherplatzbedarf: 1500Bytes.
Eine Analoguhr läßt sich mit Hilfe der im HX-20 eingebauten Digitaluhr auf das Display bringen. Zuerst muß ein Ziffernblatt aufgebaut werden. Dazu muß ein Kreis gezeichnet werden. Üblicherweise wird dies mit Hilfe der SIN- und COS-Funktion gemacht. Zu beachten ist nur, daß der HX-20 wie alle Computer mit dem amerikanischen Koordinatensystem arbeitet. Der Ursprung (0,0) befindet sich in der linken oberen Ecke. Um die Punkte für 3, 6, 9 und 12 Uhr zu setzen, läßt man die SIN- bzw. COS-Werte im Intervall /2 zeichnen. Analog dazu die ganzen Stunden mit /6.
DIE BEIDEN ZEIGER
Die Zeit muß aus dem TIME$-String über die VAL-Funktion gezogen werden. Für Stunde, Minute und Sekunde sind je zwei Variablen nötig, um einen Zeiger mit dem alten Wert zu löschen, wenn sich ein entsprechendes Zeitintervall geändert hat. Die Grafik des HX-20 ist alles andere als schnell. Aus diesem Grund mußte die Darstellung der Zeiger so einfach wie möglich gehalten werden. Es ist auch nicht möglich, den Stundenzeiger wie bei einer richtigen Analoguhr mitwandern zu lassen, und zugleich jede Sekunde anzeigen zu lassen.Voraussetzung für den korrekten Ablauf des Programms ist: Die HX-20-Uhr muß richtig gestellt werden. Stimmt die Zeit nicht, so kann man die Änderung im Direct-Mode vornehmen, zum Beispiel mit TIME$="12 : 10 : 00 ".

Alexander Broziat

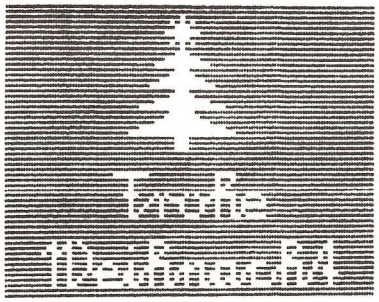



```
1000 'UHR
1010 'für Chip-Special
1011 'von Alexander Broziat
1015 'WIDTH 80,20,1
1020 DEFINT S,M          'ST,MIN,SEC
0,SEC1
1030 DEFSNG R,W,X,Y,P    'PI
1040 DEFSTR D            'DMY
1050 '
1060 CLS:PI=3.14159
1070 '
1080 'Ziffernblatt
1090 R=15
1100 FOR W=0 TO 2*PI STEP PI/120
1110   X=CINT(SIN(W)*R+60)
1120   Y=CINT(COS(W)*R+16)
1130   PSET(X,Y)
1140 NEXT W
1150 R=14
1160 FOR W=0 TO 2*PI STEP PI/2
1170   X=CINT(SIN(W)*R+60)
1180   Y=CINT(COS(W)*R+16)
1190   PSET(X,Y)
1200 NEXT W
1210 R=13
1220 FOR W=0 TO 2*PI STEP PI/6
1230   X=CINT(SIN(W)*R+60)
1240   Y=CINT(COS(W)*R+16)
1250   PSET(X,Y)
1260 NEXT W
1270 'Uhr
1280 ST0=VAL(MID$(TIME$,1,2)):IF ST
>12 THENST=ST-12
1290 MIN0=VAL(MID$(TIME$,4,2))
1300 SEC0=VAL(MID$(TIME$,7,2))
1310 'Sec.Zeiger setzen
1320   LINE(60,15)-(CINT(-SIN(PI/30
*-SEC0)*11+60),CINT(-COS(PI/30*-SEC
0)*11+15)),PSET
1330 SEC1=SEC0
1340 SEC0=VAL(MID$(TIME$,7,2)):IF S
EC0=SEC1 THEN GOTO1340
1350 LOCATE 0,0:PRINT LEFT$(TIME$,2
);:LOCATE 2,2:PRINT MID$(TIME$,4,2)
;:LOCATE 17,3:PRINT              RIGH
T$(TIME$,2);CHR$(23);
1355 'Sec.Zeiger löschen
1360 LINE(60,15)-(CINT(-SIN(PI/30*-
SEC1)*11+60),CINT(-COS(PI/30*-SEC1)
*11+15)),PRESET
1405 'Sec.Zeiger neu setzen
1410 LINE(60,15)-(CINT(-SIN(PI/30*-
SEC0)*11+60),CINT(-COS(PI/30*-SEC0)
*11+15)),PSET
1420 'Min.Zeiger setzen
1430 LINE(60,15)-(CINT(-SIN(PI/30*-
MIN0)*10+60),CINT(-COS(PI/30*-MIN0)
*10+15)),PSET
1440 MIN1=MIN0
1450 MIN0=VAL(MID$(TIME$,4,2)):IF M
IN0=MIN1 THEN GOTO1480
1455 'Min Zeiger löschen
1460 LINE(60,15)-(CINT(-SIN(PI/30*-
MIN1)*10+60),CINT(-COS(PI/30*-MIN1)
*10+15)),PRESET
1465 'Min.Zeiger neu setzen
1470 LINE(60,15)-(CINT(-SIN(PI/30*-
MIN0)*9+60),CINT(-COS(PI/30*-MIN0)*
9+15)),PSET
1480 'St.Zeiger setzen
1490 LINE(60,15)-(CINT(-SIN(PI/6*-S
T0)*8+60),CINT(-COS(PI/6*-ST0)*8+15
)),PSET
1500 ST1=ST0
1510 ST0=VAL(MID$(TIME$,1,2)):IF ST
>12 THENST=ST-12
1520 IF ST1=ST0 THENGOTO1330
1525 'St.Zeiger löschen
1530 LINE(60,15)-(CINT(-SIN(PI/6*-S
T1)*8+60),CINT(-COS(PI/6*-MIN1)*8+1
5)),PRESET
1535 'St.Zeiger neu setzen
1540 LINE(60,15)-(CINT(-SIN(PI/6*-S
T0)*8+60),CINT(-COS(PI/6*-ST0)*8+15
)),PSET:GOTO1330
```

Weihnachtsgruß

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 1600 Bytes.
Jedes beliebige Bild läßt sich mit dem HX20 ausdrucken, indem man Zeile für Zeile kodiert. Wer das Eintippen dieses Programms nicht scheut, wird mit einem Ausdruck belohnt, der zur Jahreszeit paßt. Das Punktmuster der drittuntersten

Zeile sieht beispielsweise so aus:
xxxxxx/xxxxxx/xxxxxx/..xxxx/....xx/
xxxxxx/....../....xx/....xx/xx..xx/
xx..../xxxx../xxxx../..xx../..xx../
..xxxx/....../..xxxx/..xxxx/....xx/
xxxxxx/..xxxx/xxxxxx/xxxxxx/
Dabei stellt 'x' einen gesetzten Punkt dar; '.' ist dagegen eine Leerstelle (Blank). Eine Zeile enthält 144 Punkte. Unterteilen Sie das Muster in Pakete zu je 6 Punkte, wie oben angedeutet. Ordnen Sie dann jedem 'x' eine 1 und jedem '.' eine 0 zu. Es ist eine Eigenart des Programms, daß Sie rückwärts kodieren müssen, d.h. Sie lesen die Punktzeile von hinten nach vorne. Sie erhalten somit:
111111/111111/111100/111111/110000/
111100/111100/000000/111100/001100/
usw. Ergänzen Sie nun jedes 6er-Paket am Anfang mit'11', bzw. '00'. Es spielt überhaupt keine Rolle, womit Sie ergänzen; diese beiden Bits werden nicht gebraucht. Sie dienen nur dazu, aus den 6 Bits ein ganzes Byte zu machen. Oben haben Sie nach dem Ergänzen also dann:
11111111/11111111/11111100/......11
111111/11110000/...usw.
Diese Zahlenreihe entspricht hexadezimal: FF,FF,FC,FF,FO,..usw.
Diese Hexzahlen tippen Sie dann in Data-Zeilen ab Zeilennummer 200 ein. Wenn Sie mehrere 'FF' zum Beispiel 'xy' viele hintereinander haben, so können Sie in die Datazeile auch bloß 'Qxy' schreiben. Die Anzahl, 'xy' muß immer eine zweistellige Hexzahl sein. Der Zwischenraum zwischen den ausgeprinteten Zeilen entsteht, wenn von der Routine 'PRTDOT' wieder in Basic zurückgesprungen wird.
Wenn Sie das Programm verwenden, um andere Bilder zeichnen zu lassen, so löschen Sie Zeile 160 sowie alle Data-Zeilen und ersetzen letztere durch Ihre eigenen. Beat Dörr



```
1   'Weihnachtsgruss
2   'für Chip-Special
3   'von Peter Addor
11 MEMSET &HA70:DEFINT I-K:J=0:ON E
RROR GOTO 29
12 FOR I=1 TO 7:READM:POKE &HA3F+I,
M:NEXT
13 DATA 206,10,71,189,255,145,57
14 J=J+1:READ D$
15 IF D$="H" THEN 27
16 IF LEFT$(D$,1)="Q" THEN 20
17 D=VAL("&H"+D$):POKE 2655-J,D
18 IF J<24 THEN 14
19 EXEC 2624:J=0:GOTO 14
20 Z$=RIGHT$(D$,2):Z=VAL("&H"+Z$)
21 IF Z>25-J THEN 24
22 FOR K=0 TO Z-1:POKE 2655-J-K,255
:NEXT
23 J=J+Z-1:GOTO 18
24 FOR K=0 TO 24-J:POKE 2655-J-K,25
5:NEXT
25 EXEC 2624
26 Z=Z-25+J:J=1:GOTO 21
27 FOR I=1 TO 32:READ A,B:SOUND A,B
:NEXT
28 LPRINT:LPRINT:LPRINT:LPRINT
29 IF ERR=4 THEN END ELSE RETURN
30 ON ERROR GOTO 0
31 DATA Q3C,F3,Q16,FC,F3,F,Q16,0,Q1
6,F0,0,3,Q16,0,Q16,FC,F3,F,Q16,F3,Q
17,0,Q16,F0,0,3,Q14
32 DATA FC,0,0,0,F,Q15,0,Q16,FC,0,F
,Q15,0,0,0,Q14,F0,0,0,0,3,Q15,0,Q16
,FC,0,F,Q14
33 DATA FC,0,0,0,F,Q12,FC,0,0,0,0,0
,F,Q13,FC,0,F,Q15,F0,0,3,Q14,FC,0,0
,0,F
34 DATA Q12,FC,0,0,0,0,0,F,Q11,0,0,
0,0,0,0,0,Q14,0,Q17,0,Q17,0,Q3B
35 DATA Q39,0,F,Q03,FC,0,0,Q10,F,F,
Q04,FC,Q11,F3,F,Q04,FC,Q11,FC,F,Q04
,FC,Q0F,F0,F,0,3,F3,F,F3,F,F0,F,Q0E
,F,F3
36 DATA F,C,F,F0,C,F3,FC,Q0F,0,3,F,
F,F,F3,FC,FC,FC,Q10,F3,F,F,F,F3,FC,
FF,C,Q0F,0,C,F,F,F0,F,FC,FF,F0,Q38
37 DATA Q32,FC,Q02,0,Q07,FC,3,Q04,C
,FC,Q05,FC,F,FC,FC,Q07,F3,F3,FC,Q02
,FC,FC,C,F,Q04,FC,F3,FF,C,Q07,FC,F3
,Q03,F3,FC
38 DATA F0,F3,Q04,FC,F3,FF,F0,Q08,3
```

```
,Q03,F3,FC,FC,Q05,FC,FC,FC,0,FF,0,F
F,0,F,3,F,F0,3,FC,FF,3,F3,FC,FC,Q05
,FC,C
39 DATA FC,FC,F,FC,F,F,F,C,3,F3,F0,
FC,FC,FC,F3,FC,FC,Q05,F0,F,C,FC,F3,
FC,F3,F,3,F,C,F3,F3,C,C,0,C,FC,FC,Q
05
40 DATA FC,FF,C,FC,F3,FC,F3,3,3,F,C
,F3,F3,C,F,FC,FF,C,FC,Q05,FC,FF,F0,
FC,FC,0,FC,C,C,F,F,3,F3,F0,F0,0,FF,
F0,FC,Q33
41 DATA H,7,3,0,1,7,2,6,2,7,2,6,2,5
,10,0,2,5,4,3,2,5,2,32,2,3,2,2,10,0
,2,6,2,33,2,6,2,8,2,7,2,6,2,5,10,0,
2,6,3,3,1,3,2,32,2,3,2,32,2,5,10
```

Zeichengenerierung

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion: Druckeranschluß möglich: EPSON FX-Serie, Speicherplatzbedarf: 6456 Bytes.

Die Eigenschaften des HX-20-Zeichengenerators lassen sich mit diesem Programm voll ausnutzen. Besonders interessant wird das Programm in Verbindung mit einem Drucker der FX Serie von Epson, da man mit Hilfe des Programms die Graphikzeichen des HX20 und den selbst entworfenen Zeichensatz in den Drucker übertragen kann. Lassen Sie sich nicht vom hohen Speicherbedarf abschrecken. Ohne REM's und SPACE's vermindert sich diese auf etwa 4000 Bytes.

PROGRAMMBESCHREIBUNG

Falls der Computer nicht nach einem einzigen Zeichen verlangt, können Dezimal-, Hexadezimalzahlen (z.B."&HAF") sowie ASC II-Zeichen (z.B. "A" = 65) eingegeben werden. Sobald Sie das Programm eingegeben und gestartet haben, erscheint der Programmkopf. Durch drücken einer beliebigen Taste kommen Sie weiter. Das Programm überprüft nun, ob der Zeiger für das erste Byte Ihres Zeichensatzes gesetzt ist, und ob die Memset dem Wert Ihrer Zeiger entspricht. Wenn dies nicht der Fall ist, werden Sie aufgefordert, den Zeiger zu setzen, und/oder der Memset wird berichtigt. Anschließend gelangen Sie in Menü 1. Von hier aus können Sie die angezeigten Punkte durch Drücken der jeweiligen Taste anwählen.

ZEICHENGENERIERUNG (Initial)

Dies ist der wichtigste Teil im Programm. Hier werden Ihre Zeichen definiert. Zuerst wird aber die momentane Belegung gezeigt. Sie haben nun die Möglichkeit, dieses Zeichen zu überschreiben, zu Verändern oder in Menü 1 zurückzukehren. Falls Sie nicht in Menü 1 zurückgekehrt sind, werden zwei Rasterfelder aufgebaut.In dem großen Rasterfeld wird gearbeitet,das kleine zeigt lediglich Ihr Zeichen in originalgröße.

Es können folgende Befehle zur Zeichenerzeugung verwendet werden: Alle Cursorsteuertasten funktionieren wie gewohnt. Diese sind: TAB HOME CLR DEL SCRN RECHTS LINKS OBEN UNTEN.

"SPACE" löscht den Punkt, wo sich der Cursor befindet, der Cursor wird einen Platz vorgeschoben. "A-Z", "a-z", "Graphikzeichen" setzen den Punkt an dem sich der Cursor befindet; der Cursor wird einen Platz vorgeschoben.

"<" An der Stelle, an der sich der Cursor befindet, wird ein Punkt gesetzt; der Cursor bewegt sich dann eine Stelle rückwärts. Das Zeichen "!" wird invers dargestellt.

"∧ C" Der automatische Cursorvorschub wird ein- oder ausgeschaltet (erleichtert die Erzeugung einer senkrechten Linie).

"∧ A" Die Zeichengenerierung wird abgebrochen, Rücksprung in Menü 1.

"RETURN" Das erzeugte Zeichen wird in den Zeichensatz übernommen. Sie haben die Möglichkeit, die einzelnen Bytes des Zeichens ausdrucken zu lassen. Anschließend kommen sie in Menü 1

Überprüfen des Zeichensatzes Versalien (Check)

Sie werden aufgefordert, Anfang- und Endwert einzugeben. Anschließend werden die gewünschten Zeichen auf dem LCD-Display oder dem Drukker ausgegeben. Sie können nun die Editierfunktionen des Computers benutzen. In Menü 1 kommen Sie zurück, wenn Sie eine Taste drücken deren ASC-II-Wert größer als 32 ist.

TAUSCH (Exchange)

Mit diesem Programmteil können Sie zwei Zeichen in Ihrem selbstdefinierten Zeichensatz austauschen.

KOPIEREN (Copy)

Wenn Sie Copy wählen, können Sie ein beliebiges Zeichen in Ihrem selbstdefinierten Zeichensatz übertragen. Nachdem Sie das Zeichen und seinen ASCII-Wert bestimmt haben, kommen Sie in den Programmteil "Initial", in welchem Sie denselben bearbeiten.

ZEIGER

Hier können Sie das erste Byte Ihres Zeichensatzes bestimmen. Dies ist wichtig, wenn Sie neben Ihrem Zeichensatz auch noch ein Maschinenprogramm im Memsetbereich verwenden. Sie können Ihren Zeichensatz verschieben, indem Sie ihn zuerst mit SAVE abspeichern, mit ZEIGER ihre Zeiger verändern und anschließend Ihren Zeichensatz mit LOAD wieder laden.

LADEN (Load)

Ein vorher abgespeicherter Zeichensatz wird vom Band geladen.

ABSPEICHERN (Save)

Der im Speicher des HX-20 vorhandene Zeichensatz wird auf Band abgespeichert.

AUSDRUCK MIT MATRIXDRUCKER

Alle Zeichen des HX-20 lassen sich mit dem Epson-Matrixdrucker FX-80 ausdrucken. Nachdem Sie das erste und letzte Zeichen, welches übertragen werden soll, bestimmt haben, beginnt der Computer mit der Übertragung. Diese läuft recht langsam ab, für den kompletten Zeichensatz benötigt man etwa zehn Minuten. Sie brauchen allerdings höchstens ein Viertel, also ASC (128)-ASC (159) und ASC (224)-ASC (225) zu übertragen, da der Rest aus dem ROM des Druckers übernommen wird. Nachteil: der Drucker ist nicht im Stande, zwischen diversen ASCII-Zeichen zu unterscheiden. Wenn Sie zum Beispiel ASC (254) übertragen, belegen Sie auch gleichzeitig ASC (145) und umgekehrt. Die Ursache ist ein Fehler im Betriebssystem des Drukkers, der die benötigten nationalen Zeichen aus ASC (128)-ASC (159) an die entsprechenden Plätze überträgt. So schlimm ist es aber nun auch wieder nicht, da nur wenige Plätze betroffen sind. Zudem ist eine fehlerlose Benutzung der Graphikzeichen oder des selbstdefinierten Zeichensatzes möglich, wenn man nur die selbst definierten Zeichen überträgt.

PROGRAMMAUFBAU

150-160: Der HX-20 besitzt einen Zeiger in &H11E und &H11F, der auf den ersten Bytes des selbstdefinierten Zeichensatzes hinzeigt. Das Programm überprüft, ob dieser Zeiger gesetzt ist. Wenn nicht, fordert er Sie auf, ihn einzugeben. Der Memsetbereich wird auf 193 Bytes höher als dieser Wert gesetzt, so daß der komplette Zeichensatz Platz hat.

1000-2000: Variablen. PL ASCII-Wert, welcher gerade bearbeitet wird

X,Y Cursorposition

WE Eingabewert

HX Cursorvorschub an/aus

1240-1360: Dieser Teil ist besonders interessant, da das Betriebssystem mit Cursorsteuerung ähnlich arbeitet. Zuerst wird die Cursorposition verändert, anschließend ein Punkt gesetzt, gelöscht oder nicht verändert wie bei HOME.

1500-1580: Hier werden die Punkte in den Rastern gesetzt oder gelöscht. Die Langsamkeit kommt daher, daß BASIC die Graphik nicht in dem vom Prozessor ansprechbaren Speicher verwaltet, sondern diese über eine Schnittstelle vom LCD-Display abfragen muß.

1590-1650: Es wird geprüft, ob der Cursor in Betrieb ist. Wenn nicht, wird er eingeschaltet, sonst gelöscht. Mit Zeile 1640 wird ein Herabsetzen der Pulsiergeschwindigkeit bezweckt. Diese Routine wird von Zeile 1220 angesprungen, wenn dort kein Zeichen eingegeben wird.

1660-1750: Die einzelnen Punkte der Zeichen werden durch POINT abge-

fragt. Daraus errechnen sich sechs Werte zwischen Null und 255. Diese werden anschließend in die Speicherplätze gepoket, welche durch den Algorithmus in Zeile 1700 bestimmt sind.
2000-2410: Achten Sie genauestens auf die Semikolons, da ansonsten der Drucker ein CHR$ (10) und/oder CHR$ (13) erhält,was das Konzept durcheinander bringen würde.
3000-3140: Diese Zeilen benötigen Sie (außer den REM-Zeilen) in jedem Programmsegment. Georg Balog

```
10 'Zeichendefinition
20 'für Chip-Special
30 'von Georg Balog
100 WIDTH21,5:TITLE "ZEICHDEF":DEFI
NTA-Z
110 PRINTSTRING$(20,"*"):PRINT"*ZEI
CHDEF auf HX-20*** (C) BY  G. BALOG
 *":PRINT STRING$(20,"*");
120 SOUND15,2
130 IF INKEY$="" THEN 130
140 WIDTH20,50
150 IF PEEK(&H11E)=0 AND PEEK(&H11F
)=0 THEN 3100
160 ZEIGER=PEEK(&H11E)*256+PEEK(&H1
1F):IF PEEK(310)*256+PEEK(311)-193<
ZEIGER THEN MEMSET ZEIGER+193:GOTO
160
200 '*** Menü 1 ***
210 CLS:PRINT"1 Initial. 5 Zeiger";
"2 Check 6Load/7Save";"3 Exchange 8
 FX-80":SOUND15,2
220 PRINT"4 Copy      9 Ende ";:W$=I
NPUT$(1)
230 ON VAL(W$) GOTO 1000,300,400,21
00,3100,1900,1800,2200,5000
300 '**************
301 '*** Check ***
302 '**************
310 CLS:PRINT"Druck oder LCD (P/L)"
;:IF INPUT$(1)<>"P" THEN W=0 ELSE W
=1
320 CLS:PRINT"Ausdruck von ";:GOSUB
 3000:V=ZE:PRINT"bis ";:GOSUB 3000:
B=ZE:IF B<V THEN 320
330 FOR I=V TO B
340 IF W THEN LPRINT I;"  ";CHR$(I)
350 PRINT I;"  ";CHR$(I)
360 NEXT
370 PRINT"   ()"
380 A$=INKEY$:IF A$="" THEN 380
390 IF ASC(A$)<32 THEN PRINT A$:GOT
O 380 ELSE RUN 150
400 '***************
401 '*** Tausch ***
402 '***************
410 CLS:PRINT"Tausche ASC II Wert":
PRINT"1 ";:GOSUB 3000:A1=ZE:PRINT"2
 ";:GOSUB 3000:A2=ZE
420 IF A1<224 OR A2<224 OR A1>255 O
R A2>255 THEN GOTO 200
430 W(1)=(A1-224):W(2)=(A2-224):X(1
)=W(1)*6+ZEIGER:X(2)=W(2)*6+ZEIGER
440 FOR I=1 TO 2:FOR J=0 TO 5
450 IF ALFA THEN POKE X(I)+J,Z(I,J)
 ELSE Z(I,J)=PEEK(X(I)+J)
460 NEXT:NEXT:IF ALFA THEN RUN 150
470 FOR J=0 TO 5
480 SWAP Z(1,J),Z(2,J)
490 NEXT
500 ALFA=1:GOTO 440
1000 '************************
1001 '*** Zeichenerzeugung ***
1002 '************************
1010 QX=1
1020 CLS:PRINT"Welchen ASCII Platz"
:PRINT"wollen Sie belegen ";:GOSUB
3000:PL=ZE
1030 IF PL<224 OR PL>255 THEN CLS:P
RINT"Nur ASC-II 224-255  möglich (
)":W$=INPUT$(1):GOTO 1020
1040 CLS:PRINT"ASC II ";PL:PRINT"is
t belegt mit ";CHR$(PL);" ()";:A$=
INPUT$(1)
1050 '*** Menü 2 ***
1060 CLS:PRINT"Überschreiben => Ü":
PRINT"Verändern      => V":PRINT"Men
ü           => M":PRINT"Ihrer Wahl:"
;
1070 A$=INPUT$(1):IF A$<>"Ü" AND A$
<>"V"THEN 200
1080 GL=PL:IF A$="Ü" THEN GL=32
1090 '*** Erzeugung des Rasterfelde
s ***
1100 CLS:PRINTCHR$(GL);" Bitte wart
en ..."
1110 FOR I=0 TO 5:FOR J=0 TO 7:VE(I
+1,J+1)=POINT(I,J):NEXT:NEXT
1120 CLS:FOR X=1 TO 6:FOR Y=1 TO 8
1130 IF VE(X,Y)=0 THEN 1160
1140 FOR I=0 TO 2
1150 LINE((X-1)*4+I,(Y-1)*4)-((X-1)
*4+I,(Y-1)*4+2),PSET:NEXT I:PSET(X+
25,Y+23)
1160 PSET((X-1)*4+1,(Y-1)*4+1)
1170 NEXT Y,X
1180 X=1:Y=1
1190 '***CURSORSTEUERUNG/PUNKTERZEU
GUNG ****
```

```
1200 HX=QX
1210 WE$=INKEY$
1220 IF WE$="" THEN 1590
1230 Y1=Y:WE=ASC(WE$)
1240 IF WE>65 THEN GOSUB 1500:X=X+1
*HX:GOSUB 1430:GOTO 1210:'*** Setzt
 Punkt ***
1250 IF WE=32 THEN :GOSUB 1550:X=X+
1*HX:GOSUB 1430:GOTO 1210:'*** Lösc
ht Punkt ***
1260 IF WE=60 THEN GOSUB 1430:GOSUB
 1500:X=X-1*HX:GOSUB 1430:GOTO 1210
:'*** Setzt Punkt rückwerts ***
1270 IF WE=8 THEN :X=X-1*HX:GOSUB 1
430:GOSUB 1550:GOTO 1210:'*** Delet
e ***
1280 IF WE=28 THEN X=X+1:GOSUB 1430
:GOTO 1210'*** 1280-1310 Cursorsteu
ertasten ***
1290 IF WE=29 THEN X=X-1:GOSUB 1430
:GOTO 1210
1300 IF WE=30 THEN Y=Y-1:GOSUB 1430
:GOTO 1210
1310 IF WE=31 THEN Y=Y+1:GOSUB 1430
:GOTO 1210
1320 IF WE=11 THEN Y=1:X=1:GOSUB 14
30:GOTO 1210:'*** Home ***
1330 IF WE=9 THEN X=X+4:GOSUB 1430:
GOTO1210:'*** Tabulator ***
1340 IF WE=17 THEN Y=Y+4:GOSUB 1430
:GOTO 1210' *** 1340-1350 Scrollen
***
1350 IF WE=16 THEN Y=Y-4:GOSUB 1430
:GOTO 1210
1360 IF WE=3 THEN IF HX=1 THEN HX=0
:GOTO 1210 ELSE HX=1:GOTO 1210:'***
 Umschaltung des Cursorvorschubes *
**
1370 IF WE<>33 THEN 1390 ELSE FOR X
=1 TO 6:FOR Y=1 TO 8:A=POINT((X-1)*
4,(Y-1)*4+2):IF A THEN GOSUB 1550 E
LSE GOSUB 1500:' *** Inverse Zeiche
n ***
1380 NEXT Y,X:GOSUB 1480:GOTO 1180
1390 IF WE=12 THEN FOR X=1 TO 6:FOR
 Y=1 TO 8:GOSUB 1550:NEXT Y,X:GOSUB
 1480:GOTO 1180:' *** CLS - Funktio
n ***
1400 IF WE=13 THEN 1660 ' *** Zeich
en abspeichern mit RETURN ***
1410 IF WE=1 THEN RUN 150:' *** Zur
ück zu Menü 1
1420 GOTO 1210
1430 '*** Logikteil für Cursor ***
1440 IF X>6 THEN X=X-6:Y=Y+1
1450 IF X<1 THEN X=6:Y=Y-1
1460 IF Y>8 THEN Y=1
1470 IF Y<1 THEN Y=8
1480 FOR I=3 TO 31 STEP4:LINE(0,I)-
(24,I),PRESET:NEXT I
1490 RETURN
1500 '*** Punkt setzen in Raster **
*
1510 FOR I=0 TO 2
1520 LINE((X-1)*4+I,(Y-1)*4)-((X-1)
*4+I,(Y-1)*4+2),PSET:NEXT I:PSET(X+
25,Y+23)
1530 PSET((X-1)*4+1,(Y-1)*4+1)
1540 GOTO 1580
1550 FOR I=0 TO 2
1560 LINE((X-1)*4+I,(Y-1)*4)-((X-1)
*4+I,(Y-1)*4+2),PRESET:NEXT
1570 PSET((X-1)*4+1,(Y-1)*4+1):PRES
ET(X+25,Y+23)
1580 RETURN
1590 ' *** Cursor-blink ***
1600 XA=(X-1)*4:YA=(Y-1)*4
1610 PU=POINT(XA,YA+3)
1620 IF PU=0 THEN LINE (XA,YA+3)-(X
A+2,YA+3),PSET
1630 IF PU=1 THEN LINE (XA,YA+3)-(X
A+2,YA+3),PRESET
1640 FOR R=1 TO 150:NEXT
1650 GOTO 1210
1660 '*** Abspeichern des neuen Zei
chens ***
1670 LOCATE 5,1:PRINT"Zeichen o.k ?
":IF INPUT$(1)="J" THEN LOCATE 5,1:
PRINT"Bitte warten ":ELSE LOCATE 5,
1:PRINT SPACE$(14):GOTO 1210
1680 FOR X=0 TO 5:FOR Y=0 TO 7:A=PO
INT(X*4,Y*4+2):IF A THEN Z(X)=Z(X)+
2^Y
1690 NEXT Y,X
1700 SP=(PL-224)*6+ZEIGER
1710 CLS:PRINT"Wünschen Sie den
Ausdruck der Bytes ?(J/N)";:A$=INPU
T$(1)
1720 FOR I=SP TO SP+5
1730 POKE I,Z(I-SP):IF A$="J" THEN
LPRINTI-SP;I;Z(I-SP)
1740 NEXT
1750 RUN 150
1800 '************
1801 '*** Save ***
1802 '************
1810 GOSUB 2010
1820 A$="CAS0":CLS:INPUT"Dateiname"
;B$:A$=A$+":"+B$:IF B$=""THEN 200
1830 PRINT:PRINT"Bitte warten ...
1840 PRINTBW$:OPEN"O",#1,(A$)
1850 FOR I=ZEIGER TO ZEIGER+192
1860 PRINT#1,PEEK(I)
1870 NEXT
```

```
1880 RUN 150
1900 '*************
1901 '*** Load ***
1902 '*************
1910 GOSUB 2010
1920 A$="CAS0":CLS:INPUT"Dateiname"
:B$:A$=A$+":"+B$:IF B$=""THEN 200
1930 CLS:PRINT:PRINT"Bitte warten .
.."
1940 OPEN "I",#1,(A$)
1950 FOR I=ZEIGER TO ZEIGER+192
1960 INPUT#1,G
1970 POKE I,G
1980 NEXT
1990 RUN 150
2000 '*** Subrutine zu Casettenverw
altung ***
2010 CLS:PRINT"Soll Ich winden J/N"
:A$=INPUT$(1):IF A$<>"J" THEN RETUR
N
2020 INPUT"Zu welchen TAPCNT",TAP
2030 WIND:WIND TAP:RETURN
2100 '*************
2101 '*** Copy ***
2102 '*************
2110 CLS:PRINT"Welche Zeichen wo-":
PRINT"llen Sie copieren ";:GOSUB 30
00:QX=1:GL=ZE
2120 CLS:PRINT"Welche ASCII Platz":
PRINT"wollen Sie belegen";:GOSUB 30
00:PL=ZE:IF PL<224 OR PL>255 THEN R
UN 150
2130 GOTO 1100
2200 ' ******************************
******************************
2201 ' *** Übertragen des HX-20 Zei
chensatzes auf FX Drucker ***
2202 ' ******************************
******************************
2210 CLS:PRINT"Übertrage HX Zeichen
ab   ASC-II";:GOSUB 3000:ANF=ZE:PRI
NT"bis ASC-II ";:GOSUB 3000:EN=ZE
2220 CLS:PRINT"Welchen Wert soll ":
CHR$(ANF);"in dem Drucker haben";:G
OSUB 3000
2230 DPL=ZE
2240 OPEN"O",#1,"COM0:"
2250 WIDTH"COM0:",255
2260 PRINT#1,CHR$(27);":";CHR$(0);C
HR$(0);CHR$(0);
2270 PRINT#1,CHR$(27);"%";CHR$(1);C
HR$(0);
2280 PRINT#1,CHR$(27);"%";CHR$(1);C
HR$(0);
2290 REM PRINT#1,CHR$(27);"6";
2300 PRINT#1,CHR$(27);"&";CHR$(0);C
HR$(DPL);CHR$(DPL+EN-ANF);
2310 FOR PL=ANF TO EN
2320 CLS:PRINTCHR$(PL)
2330 FOR X=0 TO 5:FOR Y=0 TO 7
2340 T=POINT(X,7-Y)
2350 IF T=1 THEN Z(X)=Z(X)+2^Y
2360 NEXT Y,X
2370  PRINT#1,CHR$(139);
2380 FOR X=0 TO 4:PRINT#1,CHR$(Z(X)
);CHR$(0);:NEXT X:PRINT#1,CHR$(Z(5)
);
2390 FOR I=0 TO 7:Z(I)=0:NEXT I,PL
2400 PRINT#1,"Zeichensatz:":FOR I=3
2 TO 255:PRINT#1,CHR$(I);CHR$(32);:
NEXT:PRINT#1,"":CLOSE
2410 RUN 150
3000 ' ******************************
3001 ' *** Eingabe von Zeichen ***
3002 ' ******************************
3010 INPUT ZE$:IF ZE$="" THEN 3010
ELSE ZE=VAL(ZE$):IF ZE=0 THEN ZE=AS
C(ZE$)
3020 IF ZE<>INT(ZE) THEN 3010
3030 RETURN
3100 '***********************
3101 '*** Zeiger setzen ***
3102 '***********************
3110 CLS:PRINT"Bitte erste Byte
Ihrer Zeichensatz   eingeben";:GOSU
B 3000:IF ZE<2624 THEN RUN 150
3120 ZE1=ZE\256:ZE2=ZE-ZE1*256:POKE
 &H11E,ZE1:POKE &H11F,ZE2
3130 MEMSET ZE+193
3140 RUN 150
5000 WIDTH40,8:PRINT:PRINT"*******
ENDE *******":END
```


Suchen Sie ein ganz bestimmtes Programm für Ihre Problemlösung?

Wir bereiten laufend neue CHIP-Specials vor. Wenn Sie uns Ihr Problem und den Computer nennen, auf dem das Programm geschrieben werden soll, haben Sie die Chance, es in einer unserer nächsten Ausgaben von CHIP-Special zu finden. Über Ihren Vorschlag freut sich:

Die Redaktion CHIP-Special
Vogel-Verlag KG
Armin Schwarz
Schillerstr. 23a

8000 München 2
==========

HX-Basic

Programmiersprache:EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 239 Bytes. Mit diesem kleinen Programm werden alle Basic-Befehlle ausgegeben, samt den Vektoren im ROM. Der Interpreter springt dort hin, wenn er auf den entsprechenden Befehl trifft. Das Programm schreibt die Ausgabe auf das LCD-Display und - falls eingeschaltet - auch auf den Minidrukker. Verwendbar ist der Output für alle,die neugierig sind, wie der Interpreter aufgebaut ist und funktioniert. Beat Dörr

```
1 'HX-20 BASIC
2 'für Chip-Special
3 'von Beat Dörr
4 DATA-32747,-32591,78,-31995,-3216
3,42:DEFINT I-N:DEFSTR A:FOR I=1 TO
 2:READ K,L,M:FOR J=1 TO M:A="$"+HE
X$(PEEK(K)*256+PEEK(K+1))+" ":K=K+2
5 N=PEEK(L):L=L+1:A=A+CHR$(127 AND
N):IF N<128 THEN 5 ELSE PRINT A:LPR
INT A:NEXT:NEXT:END
```

HEX-Dez

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 232 Bytes.
Hexadezimal-Zahlen lassen sich mit EBASIC nicht ohne weiteres direkt eingeben und verarbeiten.Das folgende Programm berechnet die Dezimalzahl aus der Zeichenfolge. Die Größe der Zahl ist unbegrenzt und wird als Absolutzahl dargestellt. Unerlaubte Zeichen außerhalb von 0 bis 9 oder A bis F rufen eine Fehlermeldung hervor. M. Kühnlein

```
1 'HEX-DEC
2 'für Chip-Special
3 'von M. Kühnlein
4 INPUT"ZAHL (HEX)";Z$
5 Z=0:FOR I=1 TO LEN(Z$)
6 TZ$=MID$(Z$,I,1)
7 TZ=VAL("&H"+TZ$)
8 IF TZ=0 AND TZ$<>"0" THEN PRINT"F
EHLER":GOTO 4
9 Z=16*Z+TZ
10 NEXT
11 PRINT"&H";Z$;" =";Z
12 GOTO 4
```

Speicher-Dump

Programmiersprache: EBASIC. Gerätekonfiguration: HX 20, Grundversion. Speicherplatzbedarf: 800 Bytes.
Der im Monitor-Programm vorhandene Dump-Befehl D gibt ab einer bestimmten Stelle den Speicherinhalt als Hex-Code auf das Display. Doch leider werden bei diesem Befehl nur immer 15 Bytes ausgegeben, womit das Durchsehen von längeren Hex-Listings stark erschwert wird. Dem vermag dieses Programm abzuhelfen: Es gibt den Code von 16 Bytes auf Display, und zwar im Hex- und ASCII-Format. Die Bedienung ist dabei ähnlich wie im Monitor: Ab einer bestimmten Stelle wird der Code ausgegeben. Diese Stelle ist dabei als Hex-Zahl (004E..FFFF) anzugeben, mit nachfolgendem RETURN. Wenn keine Zahl eingegeben und nur RETURN gedrückt wird, wer-

den einfach die nächsten 16 Bytes ausgegeben. Der Speicher von 00 (Hex) bis 4D (hex) ist dabei geschützt. Bei einer solchen Angabe werden immer die Bytes ab 8000 (Hex) ausgegeben. Verlassen werden kann das Programm durch Eintippen des Wortes QUIT.
Das Programm ist sehr schnell, da eine Routine in Maschinensprache verwendet wird, die es automatisch in den Speicher lädt und dort verwaltet. Das Maschinenprogramm befindet sich nicht im durch MEMSET definierten Bereich für solche Routinen, sondern im Bereich für die Variablen. Dafür überschreibt es aber auch keine etwaigen anderen Maschinenprogramme des Benutzers.

Beat Dörr

```
1 ' Speicher-Dump
2 ' für Chip-Special
3 ' von Beat Dörr
4 WIDTH 20,4:PRINT"Speicher-Dump V-
4.2 RETURN Weiterfahren 'QUIT' Absc
hliessen Leer-Taste druecken";:DEFI
NT I:DEFSTR C:DIM I(84):FOR I=0 TO
84:READ C:I(I)=VAL("&H"+C):NEXT:I=0
:C=INPUT$(1):CLS:DEF USR=VARPTR(I(0
))
5 I=USR(I):PRINT CHR$(11);:LINE INP
UT C:IF C="" THEN 5 ELSE IF C<>"QUI
T" THEN I=VAL("&H"+C):GOTO 5 ELSE P
RINT CHR$(11);SPC(4);CHR$(1);:END
6 DATA FE02,70CC,2020,ED00,0808,BC0
2,7223,F7FE,0270,CC86,83A7,08E7,1CA
7,30A7,44CC,8585,ED14,ED16,ED18,ED1
A,DED7,8C00,4D23,0A3C,C60F,3A8C,004
D,3822,03CE,8000,DFD7,FC02,70C3,000
4,3C18,378D,2E32,8D2B,C620,3A8D,3CC
6,0C3A,8D37
7 DATA 38DF,D7FE,0270,8D04,8D02,8D0
0,C609,3A8D,0108,3CDE,D7EC,0008,08D
F,D738,378D,0208,3216,4444,4444,8D0
3,1784,0F8B,3081,3923,028B,07A7,000
8,39C6,083C,DED7,A600,08DF,D738,802
0,815F,2302,860E,8B20,A700,085A,26E
7,3900
```

File-Finder

Programmsprache: EBASIC. Gerätekonfiguration: HX-20, Grundversion. Speicherplatzbedarf: 608 Bytes. Dieses Programm hift manchem unglücklichen Zeitgenossen aus einer mißlichen Lage: Es druckt nämlich das Inhaltsverzeichnis der Dateien aus, die sich auf einer Mikro-Kassette befinden - samt den Bandstellen, an denen die Dateien beginnen.
Die Ausgabe geschieht zuerst auf dem LCD-Display und - falls dieser eingeschaltet ist - auch auf den eingebauten Mini-Drucker. Der entstehende Streifen Papier kann dann an den Rändern abgeschnitten oder umgefalzt werden. Er paßt genau in die Schachtel für eine Mikro-Kassette.
Das Programm druckt die Köpfe sämtlicher Dateien aus. Unabhängig davon, ob diese Daten, Maschinen- oder BASIC-Programme enthalten, oder ob die Dateien vollständig oder bereits teilweise überschrieben sind. Das hat sich als zweckmäßig erwiesen.

Beat Dörr

```
1  'FileFinder
2  'für Chip-Special
3  'von Beat Doerr
10 TITLE"FileFinder"
20 DEFINT A-Z:FMT$="#### &"
21 GOTO 35
25 MSG$="Band FileName Typ":CLS:PRI
NT MSG$
30 WIND:LPRINT MSG$:LPRINT"---- ---
----- ---"
35 CLOSE:PRINT"Searching";CHR$(23);
CHR$(13);
40 ON ERROR GOTO 75:OPEN"I",#1,"CAS
0:"
45 IF CNT>0 THEN MSG$="LeseFehler":
CNT=0:GOTO 65
50 ON ERROR GOTO 0:MSG$="":FOR IX=&
H32C TO &H336
55 MSG$=MSG$+CHR$(PEEK(IX)):NEXT
60 MSG$=LEFT$(MSG$,8)+"."+RIGHT$(MS
```

```
G$,3)
65 TAPE=((TAPCNT-25)ö10)*10:PRINT U
SING FMT$;TAPE,MSG$
70 LPRINT USING FMT$;TAPE,MSG$:GOTO
 35
75 IF ERR<>53 THEN RESUME 45
80 CNT=CNT+1:IF CNT<4 THEN RESUME 3
5
85 MSG$="BandEnde EOF":PRINT USING
FMT$;TAPCNT,MSG$
90 LPRINT"---- -------- ---":LPRINT
 USING FMT$;TAPCNT,MSG$
95 SOUND 10,10:CLOSE:END
```

While/Wend-Befehle

Programmiersprache: EBASIC, Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 151 Bytes.

Der HX-20 enthält zwar die Schlüsselwörter WHILE und WEND, doch entsteht bei deren Gebrauch jedesmal ein 'Syntax Error': die Epson-Ingenieure hatten keinen Platz mehr im ROM und mußten die dazu gehörenden Maschinenroutinen herausnehmen. Dieser Beitrag stellt nun eine Lösung vor, die den Code im RAM installiert, so daß WHILE und WEND in jedem BASIC-Programm verwendet werden können.

Wenn der BASIC-Interpreter auf WHILE oder WEND trifft, verzweigt er zu zwei Stellen im RAM, an denen jeweils ein Sprungbefehl abgelegt ist. Normalerweise zeigen diese Sprünge direkt auf die Fehlerroutine für den 'Syntax Error'. Schließt man allerdings die Disketten von Epson an den HX-20 an, wird von dort ein zusätzliches Disk-BASIC geladen, das, nebst anderen Befehlen, auch den Code für WHILE/WEND enthält, und gleich die beiden Sprünge abändert.

Doch wenn man den HX-20 abschaltet und wieder ohne Disketten einschaltet, wird das Disk-BASIC gelöscht. Eine oft praktizierte Behelfslösung, den HX-20 von den Disketten abzustecken, ohne ihn auszuschalten, hat aber gravierende Nachteile: er läßt sich nicht mehr abschalten, ohne RESET zu drücken, und wenn mal unterwegs die Hardware muckt, sind die Routinen natürlich gleich verloren.

Es ist auch kaum praktikabel, Disk-BASIC als Maschinenprogramm auf die Kassette zu speichern, um es dann von dort bei Bedarf ohne Disketten wieder zu laden, denn der Code ist mit mehr als 5 KByte dafür viel zu lang: dem Mitarbeiter im Aussendienst wären da schon längst die Kunden weggelaufen.

Eine andere Lösung, die Routinen für WHILE/WEND zu extrahieren und unabhängig vom Rest separat zu verwenden, sieht auf den ersten Blick bestechend aus, denn das wären weniger als 200 Byte, die sogar dauernd im RAM gelassen werden könnten. Doch sind diese Routinen leider ortsabhängig, also nicht verschiebbar, liegen mitten im Speicher und installieren sich auch nicht selbst.

Aller Unbill zum Trotz sind diese Probleme mit dem hier vorgestellten Maschinenprogramm gelöst. Es ist so abgefaßt, daß es sich überall im Speicher zurechtfindet, nur 151 Byte lang ist und WHILE/WEND ohne Abstriche implementiert. Mancher wird sich aber natürlich zu recht fragen, was den einen solchen Aufwand rechtfertigt.

DIE WHILE-SCHLEIFE

WHILE/WEND ist eine Schleifenkonstruktion, mit der man eine Sequenz von Befehlen beliebig oft wiederholen kann, solange eine vorgegebene Anfangsbedingung erfüllt ist. Für diese Schleife hat man wie beim FOR/NEXT auf die Zeilennummern keine Rücksicht zu nehmen. Syntax sieht so aus:

```
WHILE bedingung
  befehle
WEND
```

Da WHILE/WEND die grundlegendste aller Schleifen ist, kann man alle anderen Konstruktionen damit simulieren, etwa FOR/NEXT oder REPEAT/UNTIL, bekannt von Pascal/Modula-2.

```
FOR i=von TO bis STEP mit
  befehle
NEXT

i=von
WHILE i<=bis
  befehle
  i=i+mit
WEND


REPEAT
  befehle
UNTIL bedingung


f=0
WHILE not f
  befehle
  f=bedingung
WEND
```

Die Bedingung bei WHILE wird jedesmal getestet, ob sie wahr ist, bevor die Befehle in der Schleife ausgeführt werden. Trifft die Bedingung nicht zu, werden die Befehle übersprungen und die Schleife verlassen. Das kann schon der Fall sein, bevor die inneren Befehle auch nur einmal ausgeführt worden sind: ist die Bedingung nie wahr, wird der Schleifenkörper schon am Anfang übersprungen.

Das unterscheidet WHILE/WEND denn auch von REPEAT/UNTIL. Weil dort die Bedingung erst am Schluß der Schleife getestet wird, werden die inneren Befehle zwingend mindestens einmal ausgeführt. Da WHILE aber schon am Anfang testet, können die Schleifenbefehle auch nie ausgeführt werden.

Das gleiche gilt für FOR/NEXT. Ist der Anfangsindex schon zu Beginn größer als das Endlimit, werden die Befehle bis und mit dem entsprechenden NEXT gleich übersprungen, also gar nie ausgeführt. Ältere Implementationen von BASIC können davon abweichen, das heißt durchlaufen die FOR/NEXT-Schleife mindestens einmal.

Oder FORTRAN: in FORTRAN-IV wird der äquivalente DO-LOOP auch mindestens einmal durchlaufen, nicht so aber in der neueren Version FORTRAN-77, die an die jetzt auch in BASIC verwendete Pascalkonvention angepaßt wurde. Das hat in Grenzfällen schon oft zu Schwierigkeiten bei der Übertragung von Programmen in FORTRAN-IV auf FORTRAN-77 geführt.

VERWENDUNG IM BASIC

Ebenso wie FOR/NEXT kann WHILE/WEND beliebig tief geschachtelt werden, auch gemischt mit FOR/NEXT. Das erste WHILE gehört immer zum letzten WEND der Verschachtelung, das letzte NEXT davon immer zum ersten FOR. Beim Programmieren ist es sehr hilfreich, wenn man den inneren Teil um ein oder zwei Leerschläge einruckt, wie hier gezeigt. Auf gleicher Höhe stehende Schlüsselworte gehören dann zusammen.

```
WHILE bed1
  FOR i1=von1 TO bis1
    WHILE bed2
      FOR i2=von2 TO bis2
        doSomething
      NEXT
    WEND
  NEXT
WEND
```

Falls zu einem WHILE das entprechende WEND fehlt, entsteht ein Fehler 'WE Error' (WHILE ohne WEND) mit dem Code 24, ist aber ein WEND von seinem WHILE verlassen, reagiert der HX-20 mit einem 'WH Error' (WEND ohne WHILE), Code 25 in der Systemvariable ERR.

Mit dem folgenden BASIC-Programm werden die Routinen für WHILE/WEND an einer beliebigen Adresse abgespeichert, die der Benuzer selber bestimmen kann. Die MemSet-Grenze für Maschinenprogramme wird dabei direkt hinter die WHILE/WEND-Routinen gesetzt. Falls Sie danach aber

noch weitere Programme abgespeichert haben, so muß der MEMSET-Befehl in Zeile '190' entfernt (und dann manuell gesetzt) oder entsprechend abgeändert werden.
Wer seine Routinen in einem RAM-Bereich abspeichern will, den das System vor dem Überschreiben schützt, der muß vor dem Starten dieses Programms das Schutzflag in $7E mit 'POKE 126,128' maskieren und nach Programmende mit 'POKE 126,0' den RAM-Schutz wieder einschalten:

```
100 '*********************************
110 '*     WHILE/WEND-BEFEHLE        *
120 '*      FÜR CHIP-SPECIAL         *
130 '*       VON BEAT DÖRR           *
140 '*                               *
150 '* STARTADRESSE AUFRUFEN, DANN *
160 '* SIND WHILE/WEND INSTALLIERT *
170 '*********************************
180 'while v-1.0 / 08.84 beor
190 INPUT"StartAdresse";A:MEMSET
    A+151
200 A=PEEK(&H500)*256+PEEK(&H501)
    -151
210 FOR I=A TO A+150
220  READ H$:POKE I,VAL("&H"+H$)
230 NEXT:END
240 DATA 8D,00,38,C6,0E,3A,FF,06
250 DATA 52,C6,31,3A,FF,06,55,39
260 DATA DE,FB,DF,BF,BD,98,A5,BD
270 DATA B3,CD,8D,56,31,31,26,07
280 DATA BD,84,05,35,BF,08,AC,DC
290 DATA AC,37,36,DC,BF,37,36,DC
300 DATA FB,37,36,DE,BF,DF,FB,86
310 DATA A8,36,20,1D,C6,19,7E,84
320 DATA 33,26,53,8D,2D,26,F5,35
330 DATA BF,08,AC,DE,AC,FF,08,98
340 DATA 30,EC,03,DD,FB,EE,05,DF
350 DATA AC,BD,89,02,BD,C5,DD,26
360 DATA 0E,FE,08,98,DF,AC,31,32
370 DATA 33,DD,FB,31,31,31,31,7E
380 DATA A3,79,30,08,08,08,08,A6
390 DATA 00,80,81,26,07,C6,13,BD
400 DATA 84,05,20,F3,80,27,26,0E
410 DATA DF,91,EE,01,9C,FB,07,DE
420 DATA 91,C6,07,06,26,E9,39
```

BEISPIEL: PARSER

Leider verändert der HX-20 die beiden WHILE/WEND-Pointer, wenn man ihn abschaltet oder das Systemmenu aufruft. Deshalb muß in jedem Programm, das WHILE/WEND benutzt, am Anfang die Routine INIT aufgerufen werden, die an der selbstgewählten Startadresse liegt! Beim abschließenden Beispiel erfolgt dies am Schluß der Zeile '210' in EXEC: falls Sie eine andere Startadresse als &HA40 gewählt haben, so ist der EXEC-Operand entsprechend zu ändern.

Das Programm PARSER verwandelt einen algebraischen Infix-Ausdruck (AOS, Texas Instruments) in die Postfix-Notation (UPN, Hewlett Packard). Verwendet werden dürfen die kleinen Buchstaben (a..z), verknüpft durch die Grundoperationen (+,-,*,/) und beliebig geschachtelt mit Klammern (). Wer das über mehr als hundert Ebenen machen will, kann den Stack 'S(100)' in Zeile '190' vergrössern. Der Ausdruck in Postfix wird nach RETURN ausgedruckt. Mit dieser Taste kann das Programm auch verlassen werden.

Der Stack wird dazu verwendet, um die Operationen zwischenzuspeichern, denn es wird streng nach den algebraischen Grundsätzen übersetzt: '()' vor '*/' vor '+-'. Durch die Verschachtelung können sich die verwendeten Routinen selber aufrufen, sie sind also rekursiv. Der HX-20 ist bei seinen Subroutinen völlig flexibel: diese können sich bis zu 2000 mal selbst aufrufen, erst danach ist der GOSUB/RETURN-Stack voll.

Das Programm ist ein schönes Beispiel für die Anwendung von WHILE/WEND. Eine mögliche Implementation mit IF/THEN ist nicht annähernd so verständlich und elegant. Fehlte eigentlich nur noch, dass wie in Pascal lokale Variablen möglich sind, die nur vom umgebenden Unterprogramm verwendet werden können - und das erst noch rekursiv. Doch wird das beim Microsoft-BASIC des HX-20 leider ein Traum bleiben, man

hat sich weiterhin wie hier mit einem selbstgebastelten Stack auszuhelfen.

Beispiel zum Parser:

```
+-------------------------+
  Parser a..z +-*/ ()
  ³a+b*(c-d)/((e-f)*g-h)
  abcd-*ef-g*h-/+
  ³
  _
+-------------------------+
```

```
100 '*******************************
110 '*                             *
120 '*    WHILE/WEND-BEFEHLE       *
130 '*     FÜR CHIP- SPECIAL       *
140 '*       VON BEAT DÖRR         *
150 '*                             *
160 '*******************************
170 'parser v-1.0 / 08.84 beor
180 CLEAR 1000:CLS:DEFINT A-Z
190 DEFSTR A,C,E,M,S:DIM S(100)
200 PRINT"Parser a..z +-*/ ()";
    CHR$(23)
210 PRINT">";:C=INPUT$(1):EXEC
    &HA40
220 WHILE C<>CHR$(13):GOSUB 260:
    PRINT
230  PRINT E:E="":PRINT">";:C=INPUT
    $(1)
240 WEND:END
250 'expression
260 S(P)=M:P=P+1:GOSUB 320:P=P-1:
    M=S(P)
270 WHILE C="+" OR C="-":PRINT C;
280  A=C:C=INPUT$(1):S(P)=M:P=P+1
290  GOSUB 320:P=P-1:M=S(P):E=E+A
300 WEND:RETURN
310 'term
320 GOSUB 370
330 WHILE C="*" OR C="/":PRINT C;
340  M=C:C=INPUT$(1):GOSUB 370:E=E+M
350 WEND:RETURN
360 'factor
370 IF C="(" THEN 400
380 WHILE C<"a" OR C>"z":C=INPUT$(1)
390 WEND:E=E+C:GOTO 430
400 PRINT C;:C=INPUT$(1):S(P)=A
410 P=P+1:GOSUB 260:P=P-1:A=S(P)
420 WHILE C<>")":C=INPUT$(1):WEND
430 PRINT C;:C=INPUT$(1):RETURN
```

HEXDUMP zu WHILE/WEND

```
ØØ: 8D ØØ 38 C6 ØE 3A FF Ø6 52 C6 31 3A FF Ø6 55 39
1Ø: DE FB DF BF BD 98 A5 BD B3 CD 8D 56 31 31 26 Ø7
2Ø: BD 84 Ø5 35 BF Ø8 AC DC AC 37 36 DC BF 37 36 DC
3Ø: FB 37 36 DE BF DF FB 86 A8 36 2Ø 1D C6 19 7E 84
4Ø: 33 26 53 8D 2D 26 F5 35 BF Ø8 AC DE AC FF Ø8 98
5Ø: 3Ø EC Ø3 DD FB EE Ø5 DF AC BD 89 Ø2 BD C5 DD 26
6Ø: ØE FE Ø8 98 DF AC 31 32 33 DD FB 31 31 31 31 7E
7Ø: A3 79 3Ø Ø8 Ø8 Ø8 Ø8 A6 ØØ 8Ø 81 26 Ø7 C6 13 BD
8Ø: 84 Ø5 2Ø F3 8Ø 27 26 ØE DF 91 EE Ø1 9C FB Ø7 DE
9Ø: 91 C6 Ø7 Ø6 26 E9 39
```

Suchen Sie ein ganz bestimmtes Programm für Ihre Problemlösung?

Wir bereiten laufend neue CHIP-Specials vor. Wenn Sie uns Ihr Problem und den Computer nennen, auf dem das Programm geschrieben werden soll, haben Sie die Chance, es in einem der nächsten CHIP-Specials zu finden. Über Ihren Vorschlag freut sich:

Die Redaktion CHIP-Special
Armin Schwarz
Schillerstr. 23a, 8000 München 2

ASSEMBLER-LISTNG ZU WHILE/WEND

```
*********************************
*                               *
*  WHILE/WEND IM EPSON HX-20    *
* HD6301 ON HX-20 V-1.0 08.84   *
* COPYRIGHT 1984 BY BEAT DÖRR   *
*                               *
*                               *
*********************************

*********************************
*                               *
* DIESER CODE LÄUFT IN JEDEM    *
* SPEICHER-BEREICH DES HX-20!   *
* INIT EINMAL AUFRUFEN, DANN    *
* SIND WHILE/WEND INSTALLIERT   *
*                               *
*********************************

INIT    BSR   *+2
        PULX
        LDAB  #WHILE-INIT+2
        ABX
        STX   $652
        LDAB  #WEND-WHILE
        ABX
        STX   $655
        RTS

WHILE   LDX   $FB
        STX   $BF
        JSR   $98A5
        JSR   $B3CD
        BSR   SUB
        INS
        INS
        BNE   WHILE2
        JSR   $8405
        TXS
        STS   $8AC
WHILE2  LDD   $AC
        PSHB
        PSHA
        LDD   $BF
        PSHB
        PSHA
        LDD   $FB
        PSHB
        PSHA
        LDX   $BF
        STX   $FB
        LDAA  #$A8
        PSHA
        BRA   CONT

WEND2   LDAB  #$19
        JMP   $8433
WEND    BNE   QUIT
        BSR   SUB
        BNE   WEND2
        TXS
        STS   $8AC
        LDX   $AC
        STX   $898
        TSX
        LDD   3,X
        STD   $FB
        LDX   5,X
        STX   $AC

CONT    JSR   $8902
        JSR   $C5DD
        BNE   CONT2
        LDX   $898
        STX   $AC
        INS
        PULA
        PULB
        STD   $FB
        INS
        INS
        INS
        INS
CONT2   JMP   $A379

SUB     TSX
        INX
        INX
        INX
        INX
SUB2    LDAA  0,X
        SUBA  #$81
        BNE   SUB4
        LDAB  #$13
SUB3    JSR   $8405
        BRA   SUB2
SUB4    SUBA  #$27
        BNE   QUIT
        STX   $91
        LDX   1,X
        CPX   $FB
        TPA
        LDX   $91
        LDAB  #7
        TAP
        BNE   SUB3
QUIT    RTS
        END
```

Addition mit Text

Programmiersprache: EBASIC, Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 516 Bytes für Programm, 5541 Bytes inkl. Variablen.
Aufgabe dieses Programmes ist es, eingegebene Werte (positiv oder negativ) mit Begleittext formatiert aufzuaddieren. Auf diese Weise läßt sich beispielsweise ein Haushaltsplan erstellen. Am Schluß werden alle Werte vom Minidrucker ausgedruckt. Das Programm selbst belegt im Computer wenig Speicherplatz und kann somit meist eingespeichert bleiben. Die Anwendung dieses Programmes setzt keine besonderen Programmierkenntnisse voraus. Es arbeitet im Dialog mit dem Anwender. Die Handhabung wird dadurch recht einfach und erfordert nach einiger Übung kein Nachschlagen in der Programmbeschreibung mehr.

DAS PROBLEM

Ein Schwerpunkt für Techniker und Kaufleute, aber auch im Privatbereich, dürfte die Addition von Zahlenkolonnen sein. Mit einem normalen Tischrechner, der nur über die vier Grundrechenarten verfügt, kann diese Arbeit schnell und übersichtlich ausgeführt werden. Oft sogar mit Ausdruck als Streifen. Mit einem Computer läßt sich diese Arbeit mit einem kleinen Zusatzprogramm ähnlich ausführen. Wenn man schon ein Programm hierzu entwickelt, warum denn nicht ein Programm, das es erlaubt, zu jedem Zahlenwert einen kurzen Hinweistext mit einzugeben? Mit einem solchen Programm ist man einem normalen Tischrechner überlegen.

PROGRAMMBESCHREIBUNG

Das Programm wird nach den Eingaben mit RUN gestartet. In Adresse 40 werden mit der DIM-Anweisung 200 Plätze für je eine Texteingabe mit dem dazugehörigen Zahlenwert bereitgestellt. Dies bedeutet, daß maximal 200 Werte mit dazugehörigem Text eingegeben werden können. In Adresse 50 wird nach der Überschrift gefragt. Der Summenwert S und Schleifenzähler I werden auf 0 gesetzt. Das Programm fordert zur Eingabe von Text und Zahlen auf.
Im Bereich der Adresse 100 bis 140 werden solange neue Werte erfaßt und aufsummiert, bis ein Wert mit 0 eingegeben wird. Das Programm springt dann in die Adresse 160, bildet die Formate für Text und Zahlenwerte. Auf der Anzeige erscheint die Summe und über den Minidrucker erfolgt der Ausdruck als Kontrollstreifen (Adresse 200 bis 250).

DATENEINGABE

Nach dem Einschalten erfolgt die Anzeige: Überschrift? Ein beliebiger Text kann eigegeben werden, auch Sonderzeichen wie Komma oder Doppelpunkt. Maximal jedoch 255 Zeichen. Nach Drücken der RETURN-Taste wird zur Eingabe von Text und Zahlen aufgefordert.
In diesem Programm werden zwei Datenarten verarbeitet: Der Text als Zeichenketten (String-Variable) und die Zahlen als Variablen mit doppelter Genauigkeit. Beide Datenarten werden durch Komma getrennt in die gleiche Zeile eingegeben. Zuerst der Text, dann die Ziffern. Im Textteil dürfen keine Kommata oder Doppelpunkte verwendet werden. Vom eingegebenen Textteil werden pro Zeile nur die ersten 13 Zeichen ausgedruckt, auch wenn mehr Zeichen eingegeben wurden. Die zu verarbeitenden Zahlenwerte dürfen vor dem Dezimalpunkt acht Stellen aufweisen. Sind alle Eingaben vorgenommen, ist als letzte Zeile 0,0 einzugeben. In der Anzeige erscheint die Summe. Der Minidrucker druckt alle Daten aus. Soll nicht ausgedruckt werden, kann dieser Vorgang durch Drücken der BREAK-Taste unterbrochen werden. Für eine neue Berechnung mit RUN starten.

EINGABEFEHLER

Solange eine Eingabe noch nicht mit RETURN abgeschlossen ist, besteht die Möglichkeit, zur Korrektur in dieser Zeile:
Löschen, Einsetzen, Ersetzen
Ist eine Eingabe mit RETURN abgeschlossen, kommt es bei falscher

Eingabe von Dezimalpunkt oder Komma zu der Fehlermeldung Redo. Ein Fragezeichen fordert zur erneuten Eingabe der letzten Zeile auf, die ebenfalls mit RETURN abgeschlossen werden muß. Das Programm wird hierbei nicht unterbrochen. Die fehlerhafte Zeile wird ignoriert.
Eine falsche Ziffernfolge oder Zifferneingabe kann nach betätigter RETURN-Taste jedoch nicht mehr korrigiert werden. Ein neuer Programmstart mit RUN (bzw. PF 5) ist erforderlich. Die Möglichkeit zur Eingabe des gleichen Wertes mit negativem Vorzeichen besteht auch. Diese Eingabe erscheint dann auch auf dem Kontrollstreifen. Der RUN-Befehl löscht alle zuvor eingegebenen Variablen (String-Variablen bzw. Variablen doppelter Genauigkeit) innerhalb des Programms.

GRENZEN

Die Menge der einzugebenden Werte ist auf 200 begrenzt. Die Texteingabe für die Überschrift ist auf 255 Zeichen, der Hinweistext pro Zeile auf 13 Zeichen begrenzt. Die Verarbeitung der Zahlenwerte erfolgt mit "doppelter Genauigkeit" (16-stellig).
Der Ausdruck der Zahlenwerte erfolgt jedoch nur mit 8 Stellen vor und 2 Stellen hinter dem Dezimalpunkt. Wird ein Zahlenwert 0 eingegeben, erfolgt automatisch die Summenberechnung und der Ausdruck aller bisher eingegebenen Daten. Sollte ausnahmsweise doch ein Wert als 0,00 auf dem Kontrollstreifen erscheinen, dann sollte ein winziger Wert z.B.: 0,000001 eingegeben werden.

PRAXIS-BEISPIEL

Programm mit RUN starten. Überschrift eingeben. RETURN-Taste drücken. Text und Zahlen eingeben. RETURN-Taste drücken. Zum Beenden 0,0 eingeben. RETURN-Taste drücken. Auf der LCD-Anzeige erscheint die Summe. Der Minidrucker gibt alle Daten auf dem Kontrollstreifen aus.

Arno W. Wagner

Beispiel Telephonkosten 83

Januar	35.80
Februar	65.61
März	43.83
April	75.12
Mai	63.34
Juni	54.87
Juli	56.98
August	72.90
September	66.23
Oktober	54.86
November	42.18
Dezember	143.39
SUMME:	775.11

```
*
10 'ADD  -- Addition mit Text
20  'für Chip-Special
30  'von Arno.W. Wagner
40 CLEAR3000
50 CLS
60 DIMB#(200):DIMB$(200)
70 PRINT"ÜBERSCHRIFT  ?"
80 LINE INPUT A$
90 S=0
100 I=0
110 PRINT"EINGABE:TEXT,ZAHLEN"
120 I=I+1
130 INPUT B$(I),B#(I)
140 S#=S#+B#(I)
150 IFB#(I)=0THEN180
160 GOTO120
170 LPRINT
180 U$="ö              ö":V$="########
.##"
190 PRINT"SUMME:  "::PRINT USING V$
:S#
200 LPRINTA$
210 LPRINT
220 FOR N=1 TOI-1
230 LPRINT USING U$;B$(N)::LPRINTUS
ING V$:B#(N)
240 NEXTN
250 LPRINT"-------------------------
"
260 LPRINT"SUMME:          "::LPRINT U
SING V$:S#
270 LPRINT"=========================
"
280 LPRINT
290 LPRINT
300 LPRINT
310 LPRINT
```

Grundrechen-Teacher

Programmsprache: EBASIC. Gerätekonfiguration: HX-20, Grundversion. Speicherplatzbedarf: 635 Bytes.
Teach ist mein erstes Programm auf dem HX-20. Es stellt Aufgaben in den vier Grundrechenarten. Eingegeben werden muß zuerst die höchste Zahl, mit der man rechnen will. Dann erscheint ein Menü, in dem die entsprechende Rechenart zu wählen ist. Danach stellt das Programm zehn Aufgaben. Viel Spaß!

Beat Dörr

```
1 'TEACH
2 'für Chip-Special
3 'von Beat Dörr
4 RANDOMIZE VAL(RIGHT$(TIME$,2)):CL
S:INPUT"Maximal-Zahl":E:E=E+1
5 PRINT" 1: A+B=?  2: A-B=?":PRINT"
 3: AB=?  4: AB=?"
6 A$=INPUT$(1):IF A$="1" THEN A$="+
" ELSE IF A$="2" THEN A$="-" ELSE I
F A$="3" THEN A$="" ELSE IF A$="4"
 THEN A$="" ELSE PRINT" Tschau. ma
chs guet":END
7 J=0:FOR I=0 TO 9:C=INT(SQR(RND)*E
):B=INT(SQR(RND)*E)
8 IF A$="+" THEN D=B+C
9 IF A$="-" THEN D=B-C:IF B<=C THEN
 SWAP B,C:D=-D
10 IF A$="" THEN D=B*C
11 IF A$="" THEN D=B:B=B*C
12 PRINT B;A$;C;:INPUT"= ",A
13 IF A=D THEN PRINT"Ja":F=0:J=J+1:
GOTO 16
14 PRINT "Nein":IF F=1 THEN F=0:PRI
NT B;A$;C;"=";D:PRINT:GOTO 16
15 F=1:GOTO 12
16 NEXT:IF J=10 THEN PRINT"Bravo!";
 ELSE IF J>5 THEN PRINT" Gut!"; ELS
E PRINT" Oje!";
17 PRINT J;"Richtige":GOTO 5
```

Mein Biorhythmus

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 2235 Bytes.
Viel gefragt sind die Biorhythmen, die im Jahre 1925 von Sanitätsrat Dr. med. Wilhelm Fließ und seinen drei Mitarbeitern, Dr. Hans Schlieper, Dr. Hermann Swoboda und dem Mathematiker Dr. Richard Pfennig als Periodenlehre veröffentlicht worden sind. Nach dieser Theorie beginnen bei der Geburt des Menschen drei rhythmische Schwingungen mit einer Periodendauer von 23 Tagen (physische, körperliche Schwingung), 28 Tagen (emotionale, seelische Schwingungen) und 33 Tagen (intellektuelle Schwingung). Diese Sinus-Schwingungen sind zur Hälfte im negativen Teil. Dabei ist der Ausdruck Minusphase eigentlich irreführend, denn sie ist nicht gefährlich. Sie bedeutet lediglich die Zeit der Regeneration, des Kräftespeicherns und des Aufbaus. Während dieser Zeit werden dem Körper und dem Geist neue Kräfte zugeführt. In der Plusphase ist die Zeit der erhöhten Leistungs- und Widerstandsfähigkeit.
In umfangreichen statistischen Untersuchungen hat sich gezeigt, daß den "periodischen Tagen" (Übergang von Minus auf die Plus-Phase) und den "halbperiodischen Tagen" (Übergang von Plus auf die Minusphase) besondere Bedeutung zugemessen werden muß. An diesen Tagen ist nach der Theorie der Biorhythmen die Gefahr körperlichen oder geistigen Versagens besonders groß.
Biorhythmen werden von spezialisierten Instituten für gutes Geld verkauft. Unsere Leser können sich mit ihrem HX-20 die Biorhythmen selbst ausdrucken lassen. Eine besonders schöne Grafik enthält ein Programm, das der Autor von Dr. W. Minder aus Zürich zur Veröffentlichung zugesandt bekam. Wir haben

daran lediglich die Eingabe-Routine auf einen Standard gebracht, der vom sogenannten Julianischen Jahr ausgeht und auch die Schaltjahre korrekt berechnet.
Das Programm enthält einige nicht alltägliche Routinen. So in den Zeilen 670 bis 680 den "Modulo-Befehl": x mod y
Diese Funktion bringt den ganzzahligen Rest der Division von x /y. Sind x die Anzahl Tage seit Geburt und y=23, so bringt x mod y die bei der ganzzahligen Division mit 23 übrigbleibenden Tage. Mit anderen Worten: um wieviel Tage die körperliche Schwingung schon vorgerückt ist. Und nun viel Spaß beim "Biorhythmisieren".

Pierre Weber

```
1  'Bio-Rhythmus
2  'für Chip-Special
3  'von W. Minder und P. Herrliberg
12 DIM PH(23),EM(28),IG(33),M$(12),
T$(7)
13 CLS:FOR N=1 TO 12:READ M$(N):NEX
T 'Einlesen Monats-Namen
14 FOR N=0 TO 7:READ T$(N):NEXT 'Ei
nlesen Tageszeichen
15 DATA JAN,FEB,MRZ,APR,MAI,JUN,JUL
,AUG,SPT,OKT,NOV,DEZ,S,M,D,M,D,F,S,
S
16 PI2=6.283185
17 PH=23:EM=28:IG=33
18 FOR N=0 TO 32
19 K=PH:IF N<PH THEN GOSUB 43:PH(N+
1)=I
20 K=EM:IF N<EM THEN GOSUB 43:EM(N+
1)=I
21 K=IG:GOSUB 43:IG(N+1)=I
22 NEXT
23 GOTO 52 '(Hauptprogramm)
24 'Subroutine fuer Input
25 SOUND 15,5:PRINT"DATUM FALSCH!"
26 SOUND 8,2:INPUT"DATUM(TT.MM.JJ)"
;D$:IF D$="" THEN 26
27 IF ASC(MID$(D$,3,1)) AND ASC(MID
$(D$,6,1))<>46 THEN 25
28 JJ$=RIGHT$(D$,2):MM$=MID$(D$,4,2
):TT$=LEFT$(D$,2)
29 T=INT(VAL(TT$)):M=INT(VAL(MM$)):
J%=INT(VAL(JJ$)):M%=INT(M)
30 IF T<1 OR T>31 THEN 25
31 IF T>29 AND M=2 THEN 25
32 IF T>30 AND (M=4 OR M=6 OR M=9 O
R M=11) THEN 25
33 IF M<1 OR M>12 THEN 25
34 IF J%<0 OR J%>99 THEN 25
35 J%=J%+1900:J=J%
36 'Berechnung abgek.Jul.Datum(JD)
37 IF M%>2 THEN Y%=J%:GOTO 39
38 Y%=J%-1:M%=M%+12
39 A=INT(Y%/100):B=2-A+INT(A/4)
40 JD=INT(365.25*Y%)+INT(30.6001*(M
%+1))+T+B
41 RETURN
42 'Kurvenberechnung
43 I=CINT(9*SIN(PI2*N/K))+10
44 RETURN
45 'Zeilenaufbau
46 K$=LEFT$(K$,I-1)+I$+RIGHT$(K$,L-
I)
47 RETURN
48 'Monatslaenge
49 TE=INT(ABS(M-7.5))*.5:TE=31-(TE-
INT(TE))*2
50 IF M=2 THEN TE=29-SGN(J MOD4) 'S
chaltjahr
51 RETURN
52 '---Hauptprogramm---
53 LPRINT:LPRINT:LPRINT"****BIORHYT
HMOGRAMM****"
54 CLS:LPRINT:SOUND 8,2:INPUT"NAME"
;NA$:IF NA$="" THEN 54
55 LPRINT"FÜR ":NA$
56 GOSUB 26:G=JD 'Input-Routine
57 LPRINT"GEBURTSTAG:":T:".":M$(M):
J%
58 LPRINT"*=PHYS. =EMOT. =INTEL."
59 PRINT"STARTDATUM":GOSUB 26
60 Z=JD-G 'Differenz in Tagen
61 T0=JD-3-INT((JD-3)/7)*7 'Wochent
ag
62 PH=Z MOD PH+1:EM=Z MOD EM+1:IG=Z
 MOD IG+1
63 'Druck Titel Neuer Monat
64 LPRINT SPC(30);M$(M);" ";J
65 LPRINT" NEG                POS"
66 'Druck der Rhythmen
67 K$="--------+--------":L=LEN(K
$)
68 GOSUB 48 'Berechnung Monatslaeng
e
69 I=IG(IG)
70 I$="":GOSUB 45 'Zeilenaufbau
71 I=EM(EM)
72 I$="":GOSUB 45 'Zeilenaufbau
73 I=PH(PH)
74 I$="*":GOSUB 45 'Zeilenaufbau
75 LPRINT T$(T0);K$;T
76 IF T0<7 THEN T0=T0+1 ELSE T0=1
77 IF PH<23 THEN PH=PH+1 ELSE PH=1
```

```
78 IF EM<28 THEN EM=EM+1 ELSE EM=1
79 IF IG<33 THEN IG=IG+1 ELSE IG=1
80 IF T<TE THEN T=T+1 ELSE T=1:GOTO
 83
81 IF T=1 AND M<12 THEN M=M+1:GOSUB
 48:GOTO 64
82 GOTO 67 'Neue Zeile
83 IF M=12 THEN M=1:TE=31:J=J+1:GOT
O 64 'Titel neuer Monat
84 M=M+1:GOSUB 48:GOTO 64
```

Beispiel

```
****BIORHYTHMOGRAMM****

FÜR WERNER
GEBURTSTAG: 18 .JAN 1963      *=PHYS.  #=EMOT.  o=INTEL.

          JUN  1984                          JUL  1984
  NEG                    POS          NEG                    POS
F#---------o+--------*--|  1     S|#*-------+----o-----|  1
S#-------o---+-------*---|  2     M|*#-------+---o------|  2
S|#----o-----+-----*-----|  3     D*---#-------+o-------|  3
M|--#o---------+*---------|  4     M*------#---o+--------|  5
D|--o#-------*+----------|  5     D|*-----o#--+---------|  6
M|o----*-----+----------|  6     F|--*---o-----#--------|  7
Do---*-----#--+----------|  7     S|---o*-------+-#------|  8
Fo--*--------#----------|  8     S|--o------*--+-----#---|  9
So*----------+--#-------|  9     M|o---------*-------#---|  10
S*-----------+-----#-----|  10    Do----------+--*------#--|  11
M*o----------+-------#---|  11    Mo----------+-------*---#|  12
D|*o----------+--------#--|  12    Do----------+---------*--#  13
M|--*o---------+----------#|  13    Fo----------+----------*#  14
D|----*-------+----------#  14    S|o----------+-----------*  15
F|-------o*--+----------#  15     S|--o---------+----------#*  16
S|---------o--*---------#  16     M|---o---------+---------#*|  17
S|----------o---*-------#|  17    D|----o---------+ -------#*--|  18
M|-----------+--o---*--#--|  18    M|---------o---+-----#--*---|  19
D|-----------+----o---#*---|  19    D|---------o--+--#---*-----|  20
M|-----------+-----#o---*|  20     F|----------#*-----------|  21
D|-----------+--#-----o---*  21     S|---------#*+--o--------|  22
F|----------#---------o--*  22     S|-------*-----+---o-----|  23
S|---------#--+----------*|  23     M|----*--------+------o---|  24
S|-------#-----+--------*---o  24   D|--*----------+-------o--|  25
M|-----#-------+-------*----o  25   M|*------------+--------o-|  26
D|--#----------+-----*------o  26   D*------------+---------o|  27
M|#-----------+*-----------o  27    F*------------+----------o  28
D#----------*+------------o|  28    S#*-----------+----------o  29
F#------*-----+---------o--|  29    S|#*---------+----------o  30
S#---*-------+--------o---|  30     M|--#--*------+----------o  31
                                     D|---#-----*--+----------o|  31
```

Game of Life

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 5150 Bytes.

Gegen Ende der sechziger Jahre gab John H. Conway ein Modell eines mathematischen Objekts an, das einerseits eine Selbstdarstellung und andererseits die Lösung beliebig gestellter Aufgaben enthält. Ein solches Objekt heißt "zellulares Netz" und wurde erstmals von John v. Neumann definiert. "Game of live" von Conway ist nicht eigentlich ein Spiel, denn der "Spieler" hat nichts anderes zu tun, als sich überraschen zu lassen.

Das "Spielbrett" (die Trägermenge) ist eine in Zellen eingeteilte Ebene. Man stelle sich die Fläche eines Häuserblocks vor. Eine Zelle kann besetzt oder unbesetzt sein. Man geht aus von einer willkürlich gewählten Anfangskonfiguration besetzter und unbesetzter Zellen. Nun

wird erklärt, wie man von dieser zu einer neuen Konfiguration gelangt. Man spricht dabei von einem Generationswechsel. Dadurch erklärt sich auch der Name "game of live". Man stellt sich ein Population lebender Individuen vor, die sich vermehren oder sterben können.

SPIELREGELN

Die Regeln für einen Generationswechsel lauten wie folgt;

* Eine unbesetzte Zelle geht in eine besetzte Zelle über, falls in ihrer Umgebung genau drei besetzte Zellen existieren.
* Eine besetzte Zelle bleibt besetzt, falls in ihrer Umgebung zwei oder drei Zellen besetzt sind. Andernfalls stirbt sie, d.h. geht in eine unbesetzte Zelle über.

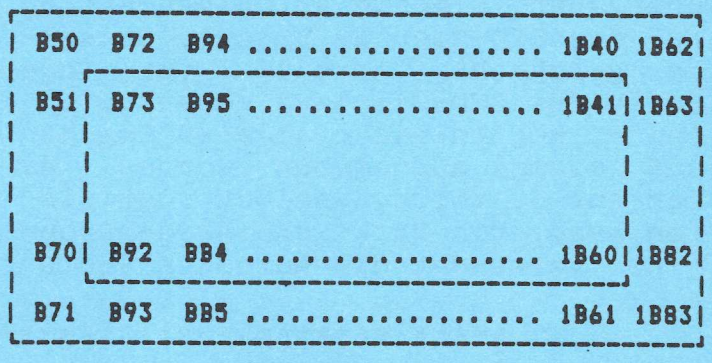
Die Umgebung einer Zelle besteht dabei aus den acht unmittelbar benachbarten Zellen, d.h. die Umgebung besteht aus einem Quadrat von 3 mal 3 Zellen.

Das Berechnen der neuen Generation ist grundsätzlich 'von Hand' möglich. Das ist jedoch eine langsame, langweilige und und sehr fehleranfällige Angelegenheit. Hier drängt sich die Hilfe des Computers geradezu auf. Für fast alle Mikrocomputer gibt es mittlerweile Programme, die das leisten.

Da diese Programme sehr rechenintensiv sind, müssen sie zumindest durch Maschinenroutinen unterstützt sein, sonst sind sie unerträglich langsam. Die meisten dieser Programme behandeln aber leider nur 'kleine Spielfelder' und zeigen diese in Low-Resolution-Manier an (24 mal 40, bestenfalls 24 mal 80 Zellen).

ALLE PUNKTE NUTZEN

Hier soll nun eine 'Life-Version' vorgestellt werden, die alle 120 mal 32 Punkte des HX-20-LCD-Displays nutzt. 'Zelle besetzt' bedeutet dann 'Punkt gesetzt' und 'Zelle unbesetzt' ist dann 'Punkt nicht gesetzt'. Ein Generationenwechsel wird in etwas mehr als einer Sekunde ausgegeben. Das ist ein recht gutes Resultat, wenn man bedenkt, daß der Computer 32*120=3840 Mal die oben angegebene Übergangsfunktion berechnen muß! Die Routine zwischen $A40 und $AD1 berechnet den Generationswechsel. Es werden die Adressen $B45 bis $B4E als Datenspeicher verwendet. Die Berechnung geschieht wie folgt: Je Bildpunkt ist ein Byte reserviert. Ist der Punkt gesetzt, so enthält das Byte eine 1, andernfalls eine 0. Natürlich würde ein Bit pro Punkt genügen. Mit einem ganzen Byte gewinnt man aber Geschwindigkeit und die Routine wird etwas einfacher. Dafür werden allerdings 12000 Bytes benötigt. Da für Randpunkte auch eine Umgebung definiert sein muß, wurde der Bildschirm mit einem Rand von einem Byte Breite 'umrahmt'. Dann wird oben links begonnen, die Speicher fortlaufend ab $B50 zu nummerieren.Diese Methode liefert einen angenehmen Nebeneffekt: Das 'Spielfeld' ist dann ein Zylinder. Was am oberen Bildrand verschwindet, taucht am unteren Bildrand wieder auf. Allerdings weist der Zylinder eine 'Schweißnaht' von zwei Bytes Breite auf. Dadurch hat man aber eine viel größere Trägermenge, als nur ein Rechteck von 120*32 Punkten.



Beispielsweise enthält die Adresse $B73 also eine 1, wenn der LCD-Punkt mit den Koordinaten (0,0) gesetzt ist. Zeigt das Indexregister nun auf irgendeine Adresse zwischen $B73 und $1B60, so steht im Akku A die Anzahl Nachbarn nach:

```
ADDA -35
ADDA -34
ADDA -33     ( Die Zahlen verstehen
ADDA -1        sich dezimal !)
ADDA 1
ADDA 33
ADDA 34
ADDA 35
```

Daraus kann entschieden werden, ob

der Punkt, auf den das Indexregister zeigt, gesetzt wird, oder nicht. Dieses Resultat schreibt die Routine in einen gleichartigen Array zwischen $1B84 und $2BB7 und überträgt nach beendeter Rechnung diesen Array wieder in den ersten.
Um die Ausgabe möglichst effizient zu gestalten, werden immer 6 mal 8 Punkte zusammengefaßt, um daraus ein eigenes Zeichen zu generieren (CHR$(224) ff). Dazu muß der Array ab $B50 so gepackt werden, daß jedem Punkt ein Bit entspricht. Dies leistet die Routine $AD2 bis $B44. Sie packt zuerst diejenigen Adressen, die den obersten 6 mal 120 Punkte des Bildschirms entsprechen und macht daraus 20 selbstdefinierte Zeichen ($E0 - $F3). Die Punktmuster sind abgelegt von $2BD0 bis $2C48, während die Initialisierungsroutine ($2C70 bis $2CA5) die Adressen $2C53 bis $2C66 mit den Charakteren $E0, $E1, ..., $F3 lädt. Im übrigen löscht diese Routine alle Speicher zwischen $B50 und $2BCF bevor überhaupt etwas geladen wird. Diese 20 Zeichen werden dann der HX-20-Betriebsroutine DSPLCN (Einsprung bei $FF49) übergeben, die sie auf einen Schlag auf das LCD-Display bringt. Dasselbe geschieht viermal, für jede Zeile des LCD-Displays separat.
Leider können nicht alle 80 Zeichen auf einmal ausgegeben werden, da man nur 32 Zeichen selbst definieren kann. Die Pack- und Ausgaberoutine verwendet die Adressen $2BC0-$2BCA als Datenspeicher. In $2BCA befindet sich ein Generationenzähler. Am Ende der Ausgaberoutine wird festgestellt, ob die Space-Taste gedrückt wurde. Ist dies der Fall, wird das Programm - nach Ausdrucken der aktuellen Generation auf dem Mikroprinter - unterbrochen.

GRUNDFIGUREN

Es gibt nun einige Grundfiguren, die der 'Life-Spieler' kennen muß. Der Blinker, eine waagerechte, oder senkrechte Reihe von drei Punkten, gehört dazu. Probieren Sie's aus.
4 Punkte, im Quadrat angeordnet, nennen sich 'Block' und verändern sich durch die Generationen hindurch nicht. Es gibt noch viele solcher stabilen Figuren: der Bienenkorb (6-Eck), die Honigfarm, der Brotlaib, etc.
Wichtige Figuren sind Gleiter und Esser: Ein Gleiter besteht z.B. aus den Punkten (60,15), (61,15), (62,15), (62,14) und (61,13).Probieren Sie das mal aus und schauen Sie, was mit dieser Figur passiert.....Verstehen Sie nun den Namen Gleiter?
Etwas ähnliches ist das Raumschiff, das aus den Punkten(10,15), (11,16), (13,16), (14,16), (14,15), (14,14) und (13,13) besteht. Dieses 'Gefährt' bewegt sich doppelt so schnell wie ein Gleiter. Zudem ist es interessant zu beachten, daß sich Gleiter nur auf 45-Grad-Bahnen bewegen, während Raumschiffe waag- und senkrecht fahren. Was geschieht

Suchen Sie ein ganz bestimmtes Programm für Ihre Problemlösung?

Wir bereiten laufend neue CHIP-Specials vor. Wenn Sie uns Ihr Problem und den Computer nennen, auf dem das Programm geschrieben werden soll, haben Sie die Chance, es in einer unserer nächsten Ausgaben von CHIP-Special zu finden. Über Ihren Vorschlag freut sich:

Die Redaktion CHIP-Special
Vogel-Verlag KG
Armin Schwarz
Schillerstr. 23a
8000 München 2

wohl, wenn zwei Raumschiffe zusammenprallen? Geben Sie obiges Raumschiff nochmals ein, zusätzlich noch ein zweites, z.B. (30,16), (30,15), (30,14), (31,13), (31,16), (32,16), §§,16), (34,15) und lassen Sie die beiden gegeneinander laufen.
Geben Sie nun den Gleiter nochmals ein, zusätzlich aber noch die Punkte (67,21), (67,22), (68,21), (68,23) (69,23), (70,23) sowie (70,24). Diese Punkte stellen einen Esser dar. Er verschlingt den Gleiter, ohne Schaden zu nehmen. Wie aber erzeugt man einen Gleiter oder gar einen Gleiterstrom? Dazu gibt es sogenannte Gleiterkanonen. Starten Sie die Programme GLIDGUN und OWNGUN und beobachten Sie, wie diese Konfigurationen alle 31 Generationen einen Gleiter 'abfeuern'. Sie müssen zuerst das Programm LIFE2.BAS geladen und mindestens einmal gestartet haben, bevor Sie mit den Programmen GLIDGUN,OWNGUN und CRASHes etwas anfangen können.
Die Frage, was passiere, wenn zwei Gleiter aufeinandertreffen, ist naheliegend. Hier geben die verschiedenen Programme CRASH1, CRASH2, ... teiweise Auskunft.
Nebst gegenseitigem Auflösen können zwei zusammenstoßende Gleiter einen Esser bauen, sich reflektieren, u.a.m.
Abschließend sei erwähnt, daß mit Esser, Gleiter und Gleiterkanonen-abstrakte Computer gebaut werden können. Haben Sie nicht Lust, selbst interessante Konfigurationen zu entdecken? Peter Addor

```
1 'LIFE
2 'für Chip-Special
3 'von Peter Addor
5 CLS:PRINT"Sie brauchen ca. 12K ":
PRINT"Bitte warten!":FOR U=0 TO 500
:NEXT
10 DEFINT A-Z:WIDTH 20.4:MEMSET &H2
D00
20 FOR T=0 TO 260:READ H$:POKE &HA4
0+T,VAL("&H"+H$):NEXT
30 FOR T=0 TO 53:READ H$:POKE &H2C7
0+T,VAL("&H"+H$):NEXT
40 EXEC &H2C70:PRINT"Input mit zu g
rossen":"Koord. abschliessen"
50 INPUT"X,Y=",X,Y:IF X<120 AND Y<3
2 THEN POKE 34*X+Y+2931,1:GOTO 50
60 CLS:PRINT"mit Space beenden":EXE
C &HB27
70 CLS:LPRINT"Abbruch nach";PEEK(&H
2BCA);"Generationen"
90 DATA CE,B,50,CC,B,73,FD,B,4A,CC,
B,73,C3,10,34,FD,B,4E,5F,EB,0,EB,1,
EB,2,EB,22,EB,24,EB,44,EB,45,EB,46,
FF,B,45,FE,B,4A,A6,0,4D,27,17,FF,B,
4A,FE,B,4E,5A,5A
91 DATA 27,7,5A,27,4,6F,0,20,19,6F,
0,6C,0,20,13,FF,B,4A,FE,B,4E,5A,5A,
5A,27,4,6F,0,20,4,6F,0,6C,0,8,FF,B,
4E,FE,B,4A,8,FF,B,4A,FE,B,45
92 DATA 8,8C,1B,3E,26,A6,CE,B,50,FF
,B,4A,CE,1B,84,A6,0,FF,B,4E,FE,B,4A
,A7,0,8,FF,B,4A,FE,B,4E,8,8C,2B,B8,
26,E9,CE,B,73,FF,B,4A
93 DATA CC,2B,D0,FD,1,1E,CE,2B,C0,6
F,6,6F,5,CC,2B,D0,ED,1,A6,6,C6,8,3D
,4F,ED,7,CC,B,58,E3,7,ED,3,EC,3,C3,
0,22,ED,3,18,C6,7,4F,AA,0,48,9,5A,2
E,F9,AA,0
94 DATA FE,2B,C1,A7,0,8,CC,2B,C0,18
,ED,1,6C,5,E6,5,C0,78,26,D8,B6,2B,C
6,CE,2C,51,A7,1,6F,0,C6,14,BD,FF,49
,CE,2B,C0,6C,6,86,4,A1,6,26,A8,6C,A
,B6,1,6D,81,20,27,3,7E,A,40,BD,FF,8
B,39
95 DATA 86,E0,B7,2B,C9,7F,2C,52,CE,
2C,53,B6,2B,C9,A7,0,7C,2B,C9,8,8C,2
C,67,26,F2,CE,B,50,6F,0,8,8C,1B,84,
26,F8,CE,1B,83,6F,0,8,8C,2B,CF,26,F
8,CE,B,73,FF,B,4A,39
```

Folgende Routinen beinhalten Grundkonfigurationen. Sie können bei Bedarf an das Grundprogramm angehängt werden.

Bitte wenden

```
1 'OWNGUN
2 'für Chip-Special
3 'von Peter Addor
4 'Laden mit MERGE zu LIFE
5 CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 21
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach":
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 42,14,42,15,41,15,47,15,48,
15,49,14,49,16,50,13,50,16,51,14,51
,16,67,16,68,17,68,18,69,17,70,18,7
1,17,72,18,75,17,76,17,76,16
```

```
5 GLIDGUN(s.WinningWays), läuft
  mit LIFE2.BIN
10 DEFINITI-Y:WIDTH20,4:EXEC&H2C70
20 FOR I=1 TO 45
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50EXEC&HA40:LPRINT"Abbruch nach";
  PEEK(&H2BCA):LPRINT"Generationen
60 DATA 10,9,10,10,11,9,11,10,21,9,
21,10,21,11,22,8,22,12,23,7,23,13,2
4,8,24,12,25,9,25,10,25,11,26,10,26
,11,26,9,31,7,31,8,31,9
70 DATA 32,6,32,7,32,9,32,10,33,6,3
3,7,33,9,33,10,34,6,34,7,34,8,34,9,
34,10,35,5,35,6,35,10,35,11,40,9,40
,10,44,7,44,8,45,7,45,8
```

```
1  'CRASH 1
2  'für Chip-Special
3  'von Peter Addor
4  'Laden mit MERGE zu LIFE
5  CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 66
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach";
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 80,11,80,12,81,12,84,13,85,
12,86,13,87,12,88,13,88,12,89,11,10
5,11,105,9,106,11,106,8,107,11,107,
9,108,10,109,10,114,10,114,9,115,10
70 DATA 10,9,10,10,11,9,11,10,21,9,
21,10,21,11,22,8,22,12,23,7,23,13,2
4,8,24,12,25,9,25,10,25,11,26,10,26
,11,26,9,31,7,31,8,31,9
80 DATA 32,6,32,7,32,9,32,10,33,6,3
3,7,33,9,33,10,34,6,34,7,34,8,34,9,
34,10,35,5,35,6,35,10,35,11,40,9,40
,10,44,7,44,8,45,7,45,8
```

```
1  'CRASH 2
2  'für Chip-Special
3  'von Peter Addor
4  'Laden mit MERGE zu LIFE
5  CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 66
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach":
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 49,10,49,9,50,9,53,8,54,9,5
5,8,56,9,57,8,57,9,58,10,74,10,74,1
2,75,10,75,13,76,10,76,12,77,11,78,
11,83,11,83,12,84,11
70 DATA 10,9,10,10,11,9,11,10,21,9,
21,10,21,11,22,8,22,12,23,7,23,13,2
4,8,24,12,25,9,25,10,25,11,26,10,26
,11,26,9,31,7,31,8,31,9
80 DATA 32,6,32,7,32,9,32,10,33,6,3
3,7,33,9,33,10,34,6,34,7,34,8,34,9,
34,10,35,5,35,6,35,10,35,11,40,9,40
,10,44,7,44,8,45,7,45,8
```

```
1  'CRASH 3
2  'für Chip-Special
3  'von Peter Adddor
4  'Laden mit MERGE zu LIFE
5  CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 66
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach";
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 51,9,51,8,52,8,55,7,56,8,57
,7,58,8,59,7,59,8,60,9,76,9,76,11,7
7,9,77,12,78,9,78,11,79,10,80,10,85
,10,85,11,86,10
70 DATA 10,9,10,10,11,9,11,10,21,9,
21,10,21,11,22,8,22,12,23,7,23,13,2
4,8,24,12,25,9,25,10,25,11,26,10,26
,11,26,9,31,7,31,8,31,9
```

```
80 DATA 32,6,32,7,32,9,32,10,33,6,3
3,7,33,9,33,10,34,6,34,7,34,8,34,9,
34,10,35,5,35,6,35,10,35,11,40,9,40
,10,44,7,44,8,45,7,45,8
```

```
1   'CRASH 4
2   'für Chip-Special
3   'von Peter Addor
4   'Laden mit MERGE zu LIFE
5   CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 66
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach";
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 52,7,52,6,53,6,56,5,57,6,58
,5,59,6,60,5,60,6,61,7,77,7,77,9,78
,7,78,10,79,7,79,9,80,8,81,8,86,8,8
6,9,87,8
70 DATA 10,7,10,8,11,7,11,8,21,7,21
,8,21,9,22,6,22,10,23,5,23,11,24,6,
24,10,25,7,25,8,25,9,26,8,26,9,26,7
,31,5,31,6,31,7
80 DATA 32,4,32,5,32,7,32,8,33,4,33
,5,33,7,33,8,34,4,34,5,34,6,34,7,34
,8,35,3,35,4,35,8,35,9,40,7,40,8,44
,5,44,6,45,5,45,6
```

```
1   'CRASH 5
2   'für Chip-Special
3   'von Peter Addor
4   'Laden mit MERGE zu LIFE
5   CLS
10 DEFINT I-Y:WIDTH 20,4:EXEC &H2C7
0
20 FOR I=1 TO 66
30 READ X,Y:POKE 34*X+Y+2931,1
40 NEXT
50 EXEC &HA40:LPRINT"Abbruch nach";
PEEK(&H2BCA):LPRINT"Generationen"
60 DATA 53,7,53,6,54,6,57,5,58,6,59
,5,60,6,61,5,61,6,62,7,78,7,78,9,79
,7,79,10,80,7,80,9,81,8,82,8,87,8,8
7,9,88,8
70 DATA 10,7,10,8,11,7,11,8,21,7,21
,8,21,9,22,6,22,10,23,5,23,11,24,6,
24,10,25,7,25,8,25,9,26,8,26,9,26,7
,31,5,31,6,31,7
80 DATA 32,4,32,5,32,7,32,8,33,4,33
,5,33,7,33,8,34,4,34,5,34,6,34,7,34
,8,35,3,35,4,35,8,35,9,40,7,40,8,44
,5,44,6,45,5,45,6
```

Code Gal

Programmiersprache: EBASIC, Gerätekonfiguration: HX 2o, Grundversion. Speicherplatzbedarf 2095 Bytes.
Zur Aufmunterung dienen diese beiden Spiele. Bei "Code" geht es darum, eine vierstellige Zahl,in der keine Ziffer doppelt und keine Null vorkommt, mit Hilfe der Angabe über die Anzahl der richtigen Ziffern (Z) und der Anzahl der richtigen Ziffern auf der richtigen Position logisch herauszufinden (ähnlich wie Superhirn).
Das Spiel "Gal" erklärt sich selbst. Das Suchwort sollte nur in GROSSBUCHSTABEN eingegeben werden.

Alexander Broziat

```
1000 'CODE-GAL
1001 'für Chip-Special
1002 'von Alexander Broziat
1010 PRINT"Galgen       ->1"
1020 PRINT"Kode Brecher->2"
1030 D$=INPUT$(1)
1040 CLS
1050 IF D$="1" THEN GOTO 1350
1060 REM PROGRAM KODE-BR
1070 '
1080 DEFINT I,J,Z,P   'INDEX,ZAHL,PS
I
1090 DEFSTR K,V,D     'KODE,VERSUCH,
DMY,DRWEI
1100 '
1110 FOR I=1 TO 4
1120   RANDOMIZE VAL(RIGHT$(TIME$,2
))
1130   DMY=MID$(STR$(INT(RND(ZAHL)*
10)),2,1)
1140   IF INSTR(DMY,"0")=0 AND INST
R(KODE,DMY)=0 THEN KODE=KODE+DMY EL
```

```
SE I=I-1
1150 NEXT I
1160 I=0
1170 ZAHL=0:PSI=0
1180   I=I+1
1190   LPRINT I;"Vs->";
1200   PRINT I;"Vs->";
1210   VERSUCH=INPUT$(4):PRINT VERS
UCH;
1220   LPRINT VERSUCH;
1230   FOR J=1 TO 4
1240     DMY=MID$(VERSUCH,J,1)
1250     IF INSTR(KODE,DMY)<>0 THEN
 ZAHL=ZAHL+1
1260     IF INSTR(KODE,DMY)=J THEN
PSI=PSI+1
1270   NEXT J
1280   PRINT"  Z";ZAHL;"P";PSI
1290   LPRINT" Z";ZAHL;"P";PSI
1300 IF VERSUCH <> KODE THEN GOTO 1
170
1310 SOUND 0,10:SOUND 7,2:SOUND 0,1
:SOUND 7,2:SOUND 0,1:SOUND 7,2:SOUN
D 0,1:SOUND 5,5
1320 PRINT I;"Versuche"
1330 LPRINT I;"Versuche"
1340 END
1350 REM PROGRAM GALGEN
1360
1370 WIDTH 20,4
1380 DEFINT I,J,Z,N,R      'ZA,NR,VE
RSUCH,RICHTIGE
1390 DEFSTR W,V            'WORT,VER
SUCH
1400
1410 PRINT"Eingabe des Wortes"
1420 INPUT WORT
1430 CLS
1440 LOCATE 0,1
1450 FOR I=1 TO LEN(WORT)
1460   PRINT" ";
1470 NEXT I
1480 ZA=ZA+1
1490   LOCATE ZA,3
1500   VERSUCH=INPUT$(1)
1510   PRINT VERSUCH;
1520 IF INSTR(WORT,VERSUCH)=0 THEN
ON ZA GOSUB 1590,1640,1690,1730,178
0,1820,1860:          GOTO 1480
1530 NR=INSTR(WORT,VERSUCH)
1540   LOCATE NR-1,1:PRINT VERSUCH
1550   RICHTIGE=RICHTIGE+1
1560   IF RICHTIGE=LEN(WORT) THEN :
SOUND 7,2:SOUND0,1:SOUND7,2:SOUND0,
1:SOUND7,2:SOUND0,1:SOUND5,5:WIDTH
80,20:END
1570   NR=INSTR(NR+1,WORT,VERSUCH):
1580 IF NR>0 THEN GOTO 1540 ELSE GO
TO 1490
1590 REM 1.ter Fehler
1600   LINE (70,31)-(119,31),PSET
1610   LINE (72,30)-(117,30),PSET
1620   LINE (74,29)-(115,29),PSET
1630 RETURN
1640 REM 2.ter Fehler
1650   LINE (76,28)-(113,28),PSET
1660   LINE (78,27)-(111,27),PSET
1670   LINE (80,26)-(109,26),PSET
1680 RETURN
1690 REM 3ter Fehler
1700   LINE (100,25)-(100,15),PSET
1710   LINE (101,25)-(101,16),PSET
1720 RETURN
1730 REM 4.ter Fehler
1740   LINE (102,18)-(102,13),PSET
1750   LINE (100,15)-(100,3),PSET
1760   LINE (101,16)-(101,2),PSET
1770 RETURN
1780 REM 5.ter Fehler
1790   LINE (100,3)-(60,3),PSET
1800   LINE (101,2)-(60,2),PSET
1810 RETURN
1820 REM 6.ter Fehler
1830   LINE (62,6)-(62,3),PSET
1840   LINE (99,10)-(80,2),PSET
1850 RETURN
1860 REM 7.ter Fehler
1870   LOCATE 10,1
1880   PRINT""
1890   LOCATE 0,1:PRINT WORT
1900   SOUND 20,5
1910   VERSUCH=INPUT$(1)
1920   WIDTH 80,20
1930 END
```


Suchen Sie ein ganz bestimmtes Programm für Ihre Problemlösung?

Wir bereiten laufend neue CHIP-Specials vor. Wenn Sie uns Ihr Problem und den Computer nennen, auf dem das Programm geschrieben werden soll, haben Sie die Chance, es in einer unserer nächsten Ausgaben von CHIP-Special zu finden. Über Ihren Vorschlag freut sich:

Die Redaktion CHIP-Special
Vogel-Verlag KG
Armin Schwarz
Schillerstr. 23a

8000 München 2
==========

Lottozahlen ermitteln

Programmiersprache: EBASIC, Gerätekonfiguration: HX20, Grundversion.
Speicherplatzbedarf: 1803 Bytes für Programm, 2685 Bytes inkl.Variablen.
Jeweils sechs Lottozahlen mit dem Zufallsgenerator erzeugt dieses Programm. Es speichert diese Zahlen ab und wertet diese nach späterem Aufrufen mit den Gewinnzahlen einschließlich Zusatzzahl aus. Alle Vorgänge werden vom Minidrucker schriftlich fixiert. Abgestimmt ist dieses Programm auf das Gewinnspiel "6 aus 49".

DAS PROBLEM

Jeder, der regelmäßig tippt und nicht immer die gleichen Zahlen verwendet, hat es nicht leicht, Zufallszahlen zu ermitteln. Sympathie zu der einen oder anderen Zahl verleiten letzten Endes immer zu einem persönlichen Schema. Ein Computer ist hier unbefangen und übernimmt die Auswertung der Gewinnzahlen. Sind mehrere Spielscheine auszuwerten, artet dies schon in Arbeit aus. Ein Computer kann dies viel schneller und zuverlässiger erledigen.

PROGRAMMBESCHREIBUNG

Das Programm wird mit RUN gestartet. Es fragt, ob Zufallszahlen erzeugt werden sollen (Z) oder ob bereits abgespeicherte Zufallszahlen vom Band eingelesen und mit den Gewinnzahlen überprüft werden sollen (P). Vor dem Erzeugen von Zufallszahlen ist die Anzahl der Spiele anzugeben. Um ein Wiederholen dieser Zahlen zu verhindern, wurde die maximale Spielanzahl auf 30 Stück begrenzt. Durch die zusätzliche Aufforderung "LEERTASTE DRÜCKEN" wird für jeden neuen Start ein individueller Vorlauf erzeugt, damit die Zufallszahlen nicht alle gleich aussehen. Dieser Vorlauf wird durch die Schleife ab Adresse 170 festgelegt. Ein Wert zwischen -32 768 und +32 767 sorgt in Adresse 320 über den Befehl RANDOMIZE für den Start der Zufallszahlen, die in Adresse 330 durch RND erzeugt werden. Diese RND-Funktion ist umhüllt von zwei Schleifen. Die innere Schleife (300 bis 400) sorgt für die 6 Zahlen pro Spiel, während die äußere für die Anzahl der Spiele insgesamt sorgt. Eine zusätzliche Schleife (340 bis 360) hinter der RND-Funktion überprüft, ob sich die gerade erzeugte Zahl auch von den vorherigen Zahlen eines jeden Spieles unterscheidet. Sind die Kriterien der äußeren Schleife (280 bis 410) erfüllt und die gewünschten Spiele ausgedruckt, wird in Adresse 930 gefragt, ob die generierten Daten abgespeichert werden sollen. Über den Minidrucker wird der Zählerstand des Laufwerks ausgedruckt. Unter dem Namen "LDATEI" erfolgt das Abspeichern der Daten (1010 bis 1090). Der erste Teil dieses Programms wäre damit beendet. Nach Eingabe von RUN oder (PF5) steht der Computer mit diesem Programm wieder zur Verfügung.
Das Überprüfen bereits abgespeicherter Daten besorgt der zweite Teil dieses Programms. Dieser Programmteil beginnt in Adresse 490 und endet mit der Adresse 910. Nachdem die Gewinnzahlen plus Zusatzzahlen eingegeben sind, wird jedes Spiel Zahl für Zahl mit den 6 Gewinnzahlen verglichen (690 bis 760). Stimmen fünf Zahlen überein, wird zusätzlich die Zusatzzahl mit allen sechs Zufallszahlen überprüft (770 bis 820). Alle Zufallszahlen, die mit den Gewinnzahlen übereinstimmen, werden formatiert ausgedruckt (730 bis 820).

DATENEINGABE

Nach dem Einschalten wird angezeigt:

Zufallszahlen: Z
Prüfprogramm-Daten vom Band: P

Wird statt Z oder P ein anderes Zeichen eingegeben, erfolgt die gleiche Anzeige erneut. Für das Erzeugen von Zufallszahlen muß Z gedückt werden. Der Computer fragt nach der Anzahl der Spiele (max. 30 Spiele, um evtl. Wiederholungen der Zufallszahlen zu vermeiden). Ist diese Eingabe größer als 30, dann wird erneut abgefragt:

```
     Anzahl der Spiele
     <= 30 !
     ?
```

Nach dem Drücken der RETURN-Taste wird aufgefordert: LEERTASTE DRÜKKEN. Der zeitliche Abstand zwischen dem Drücken von RETURN- und LEERTASTE sollte einige Sekunden betragen. Die Zufallszahlen werden spielweise ausgedruckt. Anschließend erfolgt die Anzeige:

```
        Daten speichern ?
          (J/N)
```

Vor der Abspeicherung sollte beachtet werden, daß die Kassette auf dieser Stelle frei ist. Über den Minidrucker geschieht ein Ausdruck, der die Anzahl der Spiele und die Bandzählwerkstellung enthällt. Soll mit dem Programm weiter gearbeitet werden, ist mit RUN (PF5) neu zu starten. Alle bisher ermittelten Zufallszahlen im Programm werden gelöscht. Ein Zugriff auf die Band abgespeicherten Daten ist nur mit dem Prüfprogramm möglich.

Wird das Prüfprogramm gewünscht, so ist vorher das Band auf die entsprechende Bandzählwerkstellung zu spulen: Mit dem Befehl WIND wird zum Kassettenanfang gespult und das Zählwerk automatisch auf 0 gestellt. Dann muß das Band mit dem WIND-Befehl auf die Position gebracht werden (z.B.: WIND 300).Erst jetzt kann das Prüfprogramm gestartet werden.

Folgende Anzeige erscheint:

```
        Gewinnzahlen eingeben
        1,2,3,4,5,6,Z
        ?
```

Alle Gewinnzahlen einschließlich der Zusatzzahl sind durch Komma zu trennen: Werden zuviel oder zuwenig Zahlen eingegeben, erscheint "Redo" und alle Zahlen sind neu einzugeben. Das Programm prüft nicht, ob die gleiche Gewinnzahl mehrfach vorhanden ist. Nach betätigter RETURN-Taste erfolgt der Ausdruck der Gewinnzahlen, die Zufallszahlen werden vom Band eingelesen. Anschließend werden Spielweise die Treffer ausgedruckt.

EINGABEFEHLER

Beim Abfragen können praktisch keine Eingabefehler auftreten. Werden andere Zeichen als die dargestellten eingegeben, wird die Anzeige wiederholt und ein neues Zeichen kann eingegeben werden. Nur Große Buchstaben eingeben! Bei dieser Eingabe der Gewinnzahlen kann so lange gelöscht, ersetzt oder eingesetzt werden, bis die RETURN-Taste gedrückt wird. Erscheint "Redo", muß die ganze Zeile neu eingegeben werden. Sollte das Programm aus irgend einem anderen Grund unterbrochen sein, ist ein neuer Programmstart mit RUN erforderlich.

GRENZEN

die maximale Spielzahl ist durch die DIM-Anweisung in Adresse 20 auf 30 Stück begrenzt, um eventuelle Wiederholungen der Zufallszahlen zu vermeiden. Werden mehr Spiele gewünscht, muß das Programm neu gestartet werden.

PRAXIS-BEISPIEL

Um Zufallszahlen zu ermitteln wird: Programm mit RUN gestartet. Das gewünschte Programmteil gewählt: Z. Dann: Anzahl der gewünschten Spiele eingeben. RETURN-Taste drücken. LEERTASTE drücken. Wird auf Band abgespeichert, erscheint Anzahl der Spiele und Zählwerkstellung.

Prüfprogramm für bereits auf Band abgespeicherte Daten: Kassette auf die richtige Zählwerkstellung positionieren. Programm mit RUN starten. Gewünschten Programmteil wählen: P. Gewinnzahlen mit Zusatzzahl eingeben. RETURN-Taste drücken.

Arno W. Wagner

```
1   'LOTTO
2   'für Chip-Special
3   'von A.W. Wagner
20  DIM C(30,6)
30  CLS
40  PRINT"ZUFALLSZAHLEN : Z"
50  PRINT"PRÜFPROGRAMM -"
60  PRINT"DATEN VOM BAND: P"
70  A$=INKEY$
80  IFA$="Z"THEN 110
90  IFA$="P"THEN 490
100 GOTO 70
110 CLS
120 PRINT"ANZAHL DER SPIELE"
130 PRINT"<= 30 !"
140 INPUTS
```

```
150 IF S>30 THEN 110
160 PRINT"LEERTASTE DRÜCKEN !"
170 FORZ=-32768TO32767
180 A$=INKEY$
190 IFA$=" "THEN230
200 NEXTZ
210 PRINT
220 PRINT"  BITTE WARTEN  !"
230 CLS
240 PRINT
250 PRINT"  BITTE WARTEN  !"
260 LPRINT"******  6 AUS 49  ******
"
270 LPRINT
280 FOR B=1 TO S
290 LPRINT USING"###";B;:LPRINT". *
";
300 FOR K=1TO6
310 A(K)=0
320 RANDOMIZE Z
330 X=INT(RND(1)*49)+1
340 FOR L=1TOK-1
350 IFA(L)=X THEN310
360 NEXTL
370 LPRINT USING"###";X;
380 A(K)=X
390 C(B,K)=X
400 NEXTK
410 NEXTB
420 CLS
430 LPRINT"**************************
"
440 LPRINT
450 LPRINT
460 LPRINT
470 LPRINT
480 GOTO920
490 PRINT"GEWINNZAHLEN"
500 PRINT"EINGEBEN"
510 PRINT"1,2,3,4,5,6,Z"
520 INPUT G(1),G(2),G(3),G(4),G(5),
G(6),GZ
530 LPRINT"**************************
"
540 LPRINT"GEWINNZAHLEN  6 AUS 49 :
"
550 LPRINT
560 LPRINT USING"###":G(1);G(2);G(3
);G(4);G(5);G(6);:LPRINT" /";GZ
570 LPRINT"**************************
"
580 LPRINT
590 LPRINT"AUSWERTUNG :"
600 LPRINT
610 GOSUB 1100
620 CLS
630 PRINT
640 PRINT"  BITTE WARTEN  !"
650 FOR I=1 TO S
660 T=0
670 LPRINT USING"###";I;
680 LPRINT".# ";
690 FORJ=1TO6
700 FORK=1TO6
710 IFC(I,J)=G(K)THEN 730
720 GOTO 750
730 LPRINT USING"###";G(K);
740 T=T+1
750 NEXTK
760 NEXTJ
770 IFT<>5THEN830
780 FORZ=1TO6
790 IFC(I,Z)=GZ THEN820
800 NEXT Z
810 GOTO830
820 LPRINT USING "###":GZ;
830 LPRINT
840 NEXTI
850 LPRINT"**************************
"
860 LPRINT
870 LPRINT
880 LPRINT
890 LPRINT
900 CLS
910 END
920 PRINT
930 PRINT" DATEN SPEICHERN ?"
940 PRINT"   ( J/N )"
950 A$=INKEY$
960 IFA$="J"THEN 990
970 IFA$="N"THEN END
980 GOTO 950
990 LPRINT S;"SPIELE   :";TAPCNT
1000 LPRINT
1010 OPEN"O",#1,"CAS0:LDATEI"
1020 PRINT#1,S
1030 FOR L=1 TO S
1040 FOR M=1 TO 6
1050 PRINT#1,C(L,M)
1060 NEXT M
1070 NEXT L
1080 CLOSE#1
1090 END
1100 PRINT
1110 PRINT"EINLESEN DER DATEN"
1120 PRINT"VOM BAND"
1130 OPEN"I",#1,"CAS0:LDATEI"
1140 INPUT#1,S
1150 FOR L=1 TO S
1160 FOR M=1 TO 6
1170 INPUT#1,C(L,M)
1180 NEXT M
1190 NEXT L
1200 CLOSE#1
1210 RETURN
```

Beispiel

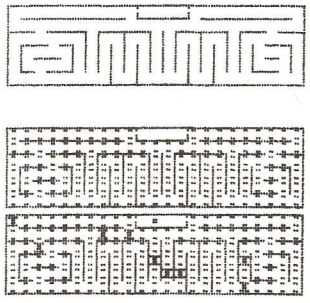
```
******  6 AUS 49  ******

  1. * 38 49 47  1 31 20
  2. * 33 28  9 41  6 25
  3. * 10 49 33 43 48  8
  4. * 25 40 15 31 38 37
  5. *  4 17 15 22 20 10
  6. * 42  4  8 17 20  1
  7. * 24 25 13 47 11 17
  8. * 44 19 40 13 33 42
  9. *  3 10 34 35 30 29
 10. * 23 43  7 26 11 28
 11. * 35 23 27 36 16 44
 12. * 39 32 18 17 36 21
 13. *  4 44 47 37 46 22
 14. * 29 26  5 14 48  7
 15. * 13 27 11 45 21  4
 16. * 42 46  6 31 12 35
 17. * 29  9 24  2 22  6
 18. * 40 30 14  2  7 31
 19. * 41 36 45 24 19 18
 20. * 34 32 46 15 49  5
 21. * 38 49 47  1 31 20
 22. * 33 28  9 41  6 25
 23. * 10 49 33 43 48  8
 24. * 25 40 15 31 38 37
 25. *  4 17 15 22 20 10
 26. * 42  4  8 17 20  1
 27. * 24 25 13 47 11 17
 28. * 44 19 40 13 33 42
 29. *  3 10 34 35 30 29
 30. * 23 43  7 26 11 28
**************************


 30 SPIELE   : 2830
```

```
**************************
GEWINNZAHLEN  6 AUS 49 :

 18 21 17 39 47  5 / 10
**************************

AUSWERTUNG :

  1.#  47
  2.#
  3.#
  4.#
  5.#  17
  6.#  17
  7.#  47 17
  8.#
  9.#
 10.#
 11.#
 12.#  39 18 17 21
 13.#  47
 14.#   5
 15.#  21
 16.#
 17.#
 18.#
 19.#  18
 20.#   5
 21.#  47
 22.#
 23.#
 24.#
 25.#  17
 26.#  17
 27.#  47 17
 28.#
 29.#
 30.#
**************************
```

PEC-Women

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 3872 Bytes.
Wer das Labyrinthspiel "Pacman" kennt, darf sich auf diese an den HX20 angepaßte Version freuen. Wer allerdings Arcadengames-Qualität erwartet, sollte auf einen für Spiele besser geeigneten Computer- übergehen. Das nette Spiele auch auf dem HX20 laufen, beweist dieses Programm.

Ralf Buchtal



```
1   'PEC-Woman
2   'für Chip-Special
3   'von Ralf Buchtal
11 TITLE"PAC-MAN":CLS:PRINT"<-  lin
ks":PRINT"->  rechts":PRINT"#   hin
auf":PRINT"RET hinab   <SPACE>"::C$
=INPUT$(1)
12 CLS:PRINT" EIN SCHAUERLICHER":PR
INT" IRRGARTEN IN EINER":PRINT" MOD
ERIGEN GRUFT..."
13 RANDOMIZE VAL(RIGHT$(TIME$,2))
14 DIM M(240),N(240)
```

```
15 FOR I=1 TO 240:READ M(I),N(I):NE
XT I
16 CLS:PRINT:PRINT"SCHWIERIGKEITSGR
AD":PRINT"(0 - 9)":C$=INPUT$(1):MO=
VAL(C$):IF MO<10 AND MO>-1 THEN 17
ELSE 16
17 CLS:MO=MO+1:SC=0:FOR Z=1 TO MO:O
(Z)=RND(1)*240:NEXT Z:GOSUB 45:LINE
(1,7)-(4,7),PRESET:GOSUB 64:X=57:Y=
2:XX=200:YY=100:GOSUB 35
18 IF SC=134 THEN SOUND 28,10:POKE
112,44:GOTO 17
19 LET XX=X:LET YY=Y
20 P=PEEK(112)
21 IF P=46 AND POINT(X+3,Y)=0 THEN
X=X+5:GOTO 26
22 IF P=47 AND POINT(X-2,Y)=0 THEN
X=X-5:GOTO 26
23 IF P=48 AND POINT(X,Y+3)=0 THEN
Y=Y+5:GOTO 26
24 IF P=44 AND POINT(X,Y-2)=0 THEN
Y=Y-5:GOTO 26
25 GOTO 27
26 GOSUB 35
27 REM
28 GOSUB 38
29 GOTO 18
30 FOR I=10 TO 1 STEP -1:SOUND I,2:
NEXT:CLS:PRINT"    VERLOREN ! ! !":
PRINT MO;". STUFE":PRINT SC;" PUNKT
E":FORI= 1 TO 1000:NEXT
31 CLS:PRINT"NEUES SPIEL":PRINT"  (
J/N) ?"
32 A$=INPUT$(1):IF A$="J" OR A$="j"
 THEN 16 ELSE IF A$<>"N" THEN 32
33 CLS:PRINT:PRINT"    TSCHÜSS ! !"
34 END
35 LINE(X,Y)-(X,Y+1),PSET:LINE(X+1,
Y)-(X+1,Y+1),PSET
36 IF POINT(X+1,Y-1)=1 THEN SC=SC+1
:SOUND 20,1
37 LINE(XX,YY-1)-(XX,YY+2),PRESET:L
INE(XX+1,YY-1)-(XX+1,YY+2),PRESET:R
ETURN
38 FOR Z=1 TO MO
39 O=O(Z):LINE(M(O),N(O))-(M(O)+3,N
(O)+3),PSET:LINE(M(O),N(O)+3)-(M(O)
+3,N(O)),PSET
40 IF M(O)+1=X AND N(O)+1=Y THEN 30
41 LINE(M(O-1),N(O-1))-(M(O-1)+3,N(
O-1)+3),PRESET:LINE(M(O-1),N(O-1)+3
)-(M(O-1)+3,N(O-1)),PRESET:IF M(O-1
)+1=X AND N(O-1)+1=Y THEN 30
42 O(Z)=O(Z)+1:IF O(Z)>240 THEN O(Z
)=2
43 NEXT Z:SOUND 1,1:RETURN
44 GOTO 16
45 LINE(0,0)-(115,0),PSET:LINE(115,
0)-(115,30),PSET
46 LINE(115,30)-(0,30),PSET:LINE(0,
30)-(0,0),PSET
47 LINE(5,5)-(45,5),PSET:LINE(50,5)
-(70,5),PSET:LINE(75,5)-(110,5),PSE
T
48 LINE(50,0)-(50,5),PSET:LINE(70,0
)-(70,5),PSET
49 LINE(0,10)-(10,10),PSET:LINE(15,
10)-(30,10),PSET:LINE(35,10)-(55,10
),PSET
50 LINE(65,10)-(85,10),PSET:LINE(90
,10)-(100,10),PSET
51 LINE(5,15)-(5,25),PSET:LINE(25,1
5)-(25,25),PSET:LINE(40,15)-(40,30)
,PSET
52 LINE(105,10)-(115,10),PSET
53 LINE(5,15)-(15,15),PSET:LINE(20,
15)-(25,15),PSET
54 LINE(5,25)-(10,25),PSET:LINE(15,
25)-(25,25),PSET:LINE(10,20)-(20,20
),PSET
55 LINE(45,10)-(45,25),PSET:LINE(50
,15)-(50,30),PSET:LINE(55,10)-(55,2
5),PSET
56 LINE(60,10)-(60,25),PSET:LINE(65
,10)-(65,25),PSET:LINE(70,15)-(70,3
0),PSET
57 LINE(75,10)-(75,25),PSET:LINE(80
,15)-(80,30),PSET:LINE(85,10)-(85,2
5),PSET
58 LINE(90,10)-(90,25),PSET:LINE(95
,15)-(95,25),PSET
59 LINE(100,15)-(110,15),PSET:LINE(
95,25)-(105,25),PSET
60 LINE(100,20)-(105,20),PSET:LINE(
110,15)-(110,25),PSET
61 LINE(30,10)-(30,25),PSET:LINE(35
,10)-(35,25),PSET
62 PRESET(50,2):PRESET(70,2)
63 RETURN
64 FOR Q=2 TO 112 STEP 5:FOR W=1 TO
 26 STEP 5:LINE(Q,W)-(Q+1,W),PSET:L
INE(Q,W+3)-(Q+1,W+3),PSET:NEXT W:SO
UND RND(1)*28,1:NEXT Q:LINE(52,1)-(
68,1),PRESET:LINE(52,4)-(68,4),PRES
ET:RETURN
65 DATA6,16,11,16,16,16,16,11,11,11
,11,6,6,6,1,6,1,1,6,1
66 DATA11,1,16,1,21,1,26,1,31,1,36,
1,41,1,46,1,46,6,51,6
67 DATA56,6,61,6,66,6,71,6,71,1,76,
1,81,1,86,1,91,1,96,1
68 DATA101,1,106,1,111,1,111,6,106,
6,101,6,96,6,91,6,86,6,81,6
69 DATA76,6,71,6,66,6,61,6,61,11,61
```

```
,16,61,21,61,26,56,26,56,21
70 DATA56.16,56,11,56,6,51,6,46,6,4
1,6,36,6,31,6,31,11,31,16
71 DATA31,21,31,26,26,26,21,26,16,2
6,11,26,11,21,16,21,21,21,21.16
72 DATA16.16,16,11,21,11,26,11,26,1
6,26,21,26,26,21,26,16,26,11.26
73 DATA6,26,1,26,1,21,1,16,1,11,6,1
1,11,11,11,6,6,6,1,6
74 DATA6,6,11,6,16,6,21,6,26,6,31,6
,36,6,41,6,46,6,51,6
75 DATA56,6,56,11,56,16,56,21,56,26
,51,26,51,21,51,16,51,11,46,11
76 DATA46,16,46,21,46,26,41,26,41,2
1,41,16,41,11,36,11,36,16,36,21
77 DATA36,26,31,26,26,26,21,26,16,2
6,11,26,11,21,6,21,6,16,11,16
78 DATA16,16,16,11,21,11,26,11,26,1
6,26,21,26,26,31,26,31,21,31,16
79 DATA31,11,31,6,36,6,41,6,46,6,51
,6,56,6,61,6,61,11,61,16
80 DATA61,21,61,26,66,26,66,21,66,1
6,66,11,71,11,71,16,71,21,71,26
81 DATA76,26,76,21,76,16,76,11,81,1
1,81,16,81,21,81,26,86,26,91,26
82 DATA96,26,101,26,106,26,106,21,1
06,16,101,16,96,16,96,11,101,11,101
,6
83 DATA96,6,91,6,86,6,86,11,86,16,8
6,21,86,26,91,26,91,21,91,16
84 DATA91,11,96,11,96,16,96,21,101,
21,106,21,106,26,111,26,111,21,111,
16
85 DATA111,11,106,11,101,11,96,11,9
6,16,96,21,101,21,106,21,106,26,101
,26
86 DATA96,26,91,26,91,21,91,16,91,1
1,96,11,101,11,101,6,96,6,91,6
87 DATA86,6,81,6,76,6,71,6,66,6,61,
6,56,6,51,6,46,6,41,6
88 DATA36,6,31,6,26,6,21,6,16,6,11,
6,11,11,16,11,16,16,11,16
```

Methusalem

Programmiersprache: EBASIC. Gerätekonfiguration: HX20 oder PX8, Grundversion. Speicherplatzbedarf: 2595 Bytes.
Dieses Programm berechnet mit gezielten Fragen und anhand statistischer Daten die theoretische Lebens-Erwartung eines Erwachsenen. Dabei kommt Angeborenes, wie Angelerntes zum tragen.Programmiert hat es Peter Addor,der Autor hat die Fragen neu formuliert und dem Display des HX-20 angepaßt. Viel Vergnügen und ein langes Leben!

Beat Dörr

```
1 'LEBEN
2 'für Chip-Special
3 'von Peter Addor & Beat
4 WIDTH 20,20:PRINT"Lebenserwartung
 für Erwachsene zwischen  20 und 65
 Jahren.     (Drücke Taste)";:PRINT
INPUT$(1):A=72
5 CLS:PRINT:PRINT"      Geschlecht
       männlich?":IF INPUT$(1)="J"
 THEN A=A-3 ELSE 7
6 CLS:PRINT"      Jedes Jahr
 in ärztliche          Kontrolle?":I
F INPUT$(1)="J" THEN A=A+1:GOTO 8 E
LSE 8
7 CLS:PRINT:PRINT"   Jedes Jahr zum
       Frauenarzt?":A=A+4:IF INPUT
$(1)="J" THEN A=A+2
8 CLS:PRINT" Wohnst Du in einer  St
adt mit mehr als     zwei Millionen
      Einwohnern?";:IF INPUT$(1)="
J" THEN A=A-2:GOTO 10
9 CLS:PRINT" Wohnst Du in einer
Kleinstadt mit     weniger als 10'00
0      Einwohnern?";:IF INPUT$(1)="
J" THEN A=A+2
10 CLS:PRINT:PRINT"  Arbeitest Du a
m      Schreibtisch?":IF INPUT$(1)=
"J" THEN A=A-3
11 CLS:PRINT"Erfordert Dein Berufsc
hwere körperliche       Arbeit?":IF
 INPUT$(1)="J" THEN A=A+3
12 CLS:PRINT"Treibst Du mehr als  5
.5 Stunden Sport       pro Woche?":I
F INPUT$(1)="J" THEN A=A+4:GOTO 14
13 CLS:PRINT:PRINT" Treibst Du 2-3
Std.  Sport pro Woche?":IF INPUT$(1
)="J" THEN A=A+2
14 CLS:PRINT:PRINT" Lebst Du mit ei
nem   festen Partner?":IF INPUT$(1)
="J" THEN A=A+5:GOTO 16
```

```
15 CLS:PRINT"Wieviele Jahrzehnte
seit Deinem 25.  Altersjahr lebst
Du   ohne Partner?";:B=VAL(INPUT$(
1)):A=A-B
16 CLS:PRINT:PRINT"Schläfst Du mehr
 als    10 Stunden?":IF INPUT$(1)="
J" THEN A=A-4
17 CLS:PRINT"   Bist Du leicht
  erregbar und          aggressiv?":
IF INPUT$(1)="J" THEN A=A-3 ELSE A=
A+3
18 CLS:PRINT:PRINT" Bist Du glückli
ch?":IF INPUT$(1)="J" THEN A=A+1 EL
SE A=A-2
19 CLS:PRINT"   Innerhalb des    le
tzten Jahres eine    Strafe für zu
    schnelles Fahren?";:IF INPUT$(1
)="J" THEN A=A-1
20 CLS:PRINT:PRINT"  Verdienst Du m
ehr als 100'000 DM/Jahr?";:IF INPUT
$(1)="J" THEN A=A-2
21 CLS:PRINT"     Bildung:
 1 Hochschule          2 Höhere Schul
e     3 Sonstige?";:B=VAL(INPUT$(1)
):IF B<3 THEN A=A+B
22 CLS:PRINT:PRINT"Bist Du über 65
und   arbeitest noch?":IF INPUT$(1)
="J" THEN A=A+3
23 CLS:PRINT"   Sind alle vier
 Grosseltern 80          geworden?":
IF INPUT$(1)="J" THEN A=A+6
24 CLS:PRINT"  Ist einer Deiner
 Grosseltern 85          geworden?":
IF INPUT$(1)="J" THEN A=A+2
25 CLS:PRINT" Ist ein Elternteil  u
nter 50 an einem   Schlag- oder Her
z-  Anfall gestorben?";:IF INPUT$(1
)="J" THEN A=A-4
26 CLS:PRINT" Hatten Geschwister od
er Eltern unter 50Herzstörungen, Kr
ebs   oder Diabetes?";:IF INPUT$(1)
="J" THEN A=A-3
27 CLS:PRINT" Rauchen: Pack/Tag   1
  > 2    2  1-2    3  0.5-1  4  Nic
ht?":B=VAL(INPUT$(1)):A=A+3*B-12-(
B=1)
28 CLS:PRINT"  Trinkst Du soviel  A
lkohol pro Tag wie  in einem vierte
l   Liter Schnaps ist?";:IF INPUT$(
1)="J" THEN A=A-1
29 CLS:PRINT"    Übergewicht:    1
> 22kg  2 13-22kg 3 5-13kg  4 nicht
s?":B=VAL(INPUT$(1)):A=A+2*B-8+2*(B
=1)
30 CLS:PRINT"       Alter:       1
20-40    2 40-50  3 50-60    4 > 60
?":B=VAL(INPUT$(1)):A=A+B+1
31 CLS:PRINT USING"   Deine Lebens-
     erwartung liegt      bei ## J
ahren";A:END
```

Springer

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 750 Bytes.
Schachfreunde werden sich über diesen Leckerbissen freuen: Das Programm findet die Tour eines Springers auf einem Schachbrett, wobei jedes Feld genau einmal angesprungen wird. Das Programm verlangt die Eingabe der Größe des Brettes (5 bis 32) sowie die Koordinaten des Startfeldes. Die Springer-Tour wird nach der Berechnung als Graphik auf das Display gegeben. Beat Dörr

```
1 'SPRINGER
2 'für Chip-Special
3 'von Beat Dörr
4 CLS:DEFINT A-Z:DEF FN T(A,B)=A<1
OR A>N OR B<1 OR B>N:PRINT"Springer
"
5 LOCATE 0,1:INPUT"n";N:IF N<5 OR N
>32 THEN 5
6 LOCATE 0,2:INPUT"i,j";I,J:IF FN T
(I,J) OR N/2<>N÷2 AND (I+J)/2<>(I+J
)÷2 THEN 6
7 I1=I:J1=J:GOSUB 8:I=I1:J=J1:F(I,J
)=1:PSET(I+87,J-1):SOUND 10,1:GOSUB
 11:FOR Z=2 TO N*N:K1=0:GOSUB 13:IF
 K1=0 THEN END ELSE I=I+X(K1):J=J+Y
(K1):F(I,J)=Z:PSET(I+87,J-1):GOSUB
11:NEXT:END
8 DATA-2,1,-1,2,1,2,2,1,2,-1,1,-2,-
1,-2,-2,-1:OPTION BASE 1:DIM F(N,N)
,X(8),Y(8):FOR K=1 TO 8:READ X(K),Y
(K):NEXT
9 FOR I=1 TO N:FOR J=1 TO N:FOR K=1
```

```
 TO 8:U=I+X(K):V=J+Y(K):IF FN T(U,V
) THEN ELSE F(U,V)=F(U,V)-1
10 NEXT:NEXT:NEXT:RETURN
11 FOR K=1 TO 8:U=I+X(K):V=J+Y(K):I
F FN T(U,V) THEN ELSE IF F(U,V)<0 T
HEN F(U,V)=F(U,V)+1
12 NEXT:RETURN
13 M=-10:FOR K=1 TO 8:U=I+X(K):V=J+
Y(K):IF FN T(U,V) THEN 15
14 IF F(U,V)<=0 AND F(U,V)>M THEN M
=F(U,V):K1=K
15 NEXT:RETURN
```

Loeillet

Programmiersprache: EBASIC. Gerätekonfiguration: HX20, Grundversion. Speicherplatzbedarf: 1638 Bytes.
Ein Musikstück des gleichnamigen Komponisten läßt sich mit diesem Programm spielen. Wenn Sie es eingetippt und zum Laufen gebracht haben, werden Sie sich die Frage selbst beantworten können: Wie gut eignet sich der HX20 für Musik?

Peter Addor und Beat Dörr

```
1 'LOEILLET
2 'für Chip-Special
3 'von Peter Addor
4 CLS:PRINT TAB(4);CHR$(135);STRING
$(10,133);CHR$(136);TAB(4);CHR$(134
);" Loeillet ";CHR$(134);TAB(4);CHR
$(137);STRING$(10,133);CHR$(138);CH
R$(23);:DEFINT A-Z:FOR I=1 TO 303:R
EAD A,B:SOUND A,B:NEXT:END
5 DATA 6,4,8,2,10,2,13,4,40,4,13,8,
14,4,15,4,40,4,13,4,10,2,9,2,8,8,10
,4,11,4,9,2,8,2,7,4,40,4,13,4,6,4,8
,4,9,4,10,4,11,2,7,2,7,6,6,2,6,8,6,
4,8,2,10,2,13,4,40,4,13,4,14,4,15,4
,40,4,13,4,10,2,9,2,8,4,8,4,9,6,10,
1,11,1,11,6,11,2,12,1,11,1,12,1,11,
1
6 DATA 10,4,13,2,14,2,40,2,10,2,13,
2,14,2,15,1,14,1,15,1,14,1,15,1,14,
1,13,2,13,8,14,4,15,4,14,4,13,4,40,
4,13,4,14,2,15,2,12,6,11,2,10,4,11,
4,13,1,12,1,13,1,12,1,13,1,12,7,11,
2,10,2,12,1,11,1,12,1,11,1,12,1,11,
7,10,2,9,2,11,1,10,1,11,1,10,1,11,1
7 DATA 10,7,11,2,10,2,9,6,8,2,7,4,5
,4,9,1,8,1,9,1,8,7,10,2,9,2,8,2,9,2
,11,2,10,2,9,2,10,4,11,4,12,4,13,4,
9,2,11,2,10,2,9,2,10,1,9,1,10,1,9,1
,10,1,9,1,8,2,8,4,8,4,10,2,12,2,15,
4,14,4,15,4,12,2,11,2,10,4,11,2,9,2
,11,1,10,1,11,1,10,7,13,2,12,2,39,4
,9,4
8 DATA 12,4,7,4,9,1,8,1,9,1,8,3,10,
2,9,2,8,2,7,4,8,4,9,4,10,4,6,4,9,6,
8,2,7,2,6,2,7,4,39,2,12,2,7,1,6,1,7
,1,6,1,7,1,6,1,5,2,5,8,6,4,7,2,8,2,
9,2,8,2,7,4,5,4,6,4,9,2,10,2,9,2,8,
2,7,2,6,2,7,4,10,2,11,2,10,2,9,2,8,
2,7,2,8,4,6,4,7,4,8,4,10,4,6,4,6,4,
11,4
9 DATA 9,4,10,4,11,4,9,2,8,2,7,6,7,
2,13,1,40,1,13,1,40,5,10,4,11,2,10,
2,9,2,10,2,11,6,11,2,11,2,10,2,10,4
,10,2,9,2,9,4,9,2,8,2,8,4,15,2,14,2
,14,4,14,2,13,2,14,1,13,3,14,2,15,2
,40,2,10,2,13,2,14,2,15,1,14,1,15,1
,14,1,15,1,14,1,13,2,13,4,6,4,8,2,1
0,2
10 DATA 13,4,40,4,14,1,13,1,14,1,13
,1,14,1,13,1,14,1,13,1,14,4,15,4,40
,4,13,4,10,2,9,3,8,1,9,1,8,5,10,4,1
1,4,9,2,8,2,7,4,40,4,14,1,13,1,14,1
,13,5,10,4,11,2,10,2,9,2,8,3,7,1,8,
1,7,1,8,1,7,1,6,2,6,16
```

****

Suchen Sie ein ganz bestimmtes Programm für Ihre Problemlösung?

Wir bereiten laufend neue CHIP-Specials vor. Wenn Sie uns Ihr Problem und den Computer nennen, auf dem das Programm geschrieben werden soll, haben Sie die Chance, es in einem der nächsten CHIP-Specials zu finden. Über Ihren Vorschlag freut sich:

Die Redaktion CHIP-Special

****

RAM- und ROM-Addressen

Wenn Sie zu den engagierten Selbstprogrammierern gehören, die aus ihrem Epson HX-20 mehr herausholen, als es mit Hilfe des Handbuches möglich ist, so wird Ihnen die nachfolgende Aufstellung wertvolle Dienste leisten. Sie finden hier eine der größten Aufstellungen an RAM-Adressen, ROM-Routinen und Adressen zur Variablen-Speicherung.

Lutz Ahrens

RAM-ADRESSEN

Ø- 77	0000-004D	GESCHUETZTER SPEICHERBEREICH in der Haupt-CPU ("protected"), kann nur gelesen oder ueberschrieben werden, wenn in Adresse $7E ein $80 steht oder vom Assembler-Programm aus. Vorsicht mit den Adressen 44 und 45 ($2C und $2D), hierbei erfolgt oft ein System-Crash und es muss neu initialisiert werden!!!
Ø	0000	Direction Register Port 1 (RS-232, etc.).
1	0001	Direction Register Port 2 (Serial, RS-232, Bar Code).
2	0002	Port 1 (RS-232, etc.).
3	0003	Port 2 (Serial, RS-232, Bar Code).
4	0004	Direction Register Port 3 (AØ...A7, DØ...D7).
5	0005	Direction Register Port 4 (A8...A15).
6	0006	Port 3 (AØ...A7, DØ...D7).
7	0007	Port 4 (A8...A15).
8	0008	Timer Control.
9- 10	0009-000A	Free Running Counter CPU R/W.
11- 12	000B-000C	Output Compare Register R/W.
13- 14	000D-000E	Input Capture Register READ.
15	000F	P3 Control Register.
16	0010	Serial Speed Rate.
17	0011	Serial Control Status.
18	0012	Receive Data Register.
19	0013	Transmission Data Register.
20	0014	RAM Control: External RAM/Internal RAM Switch
21- 31	0015-001F	nicht benutzt (normal 'FF').
32	0020	Keyboard Scan KSC Ø...7 Outputs/SW6 Read
34	0022	Keyboard Inputs KRTN Ø...7.
38	0026	Cardridge Interface/Interruption Mask Release/LCD Chip Select/Key Mask.
40	0028	Keyboard Inputs KRTN 8...9, PWSW, BUSY (SO).
42	002A	Serial Clock Generated by ANDing R/W Signal.
64- 73	0040-0049	REGISTER DER EINGEBAUTEN UHR. Die Zahlen werden im BCD-Format abgelegt. Das Datum '03/29/83' ergibt in den entsprechenden Speicherstellen hexadezimal '29 03 83'.
64	0040	Sekunden.
65	0041	Sekunden (Alarm, normal 'FF').
66	0042	Minuten.
67	0043	Minuten (Alarm, normal 'FF').
68	0044	Stunden.
69	0045	Stunden (Alarm, normal 'FF').
70	0046	Wochentag.

71	0047	Tag.
72	0048	Monat.
73	0049	Jahr.
74	004A	Enthaelt $2F, bei kleineren Werten geht die Uhr langsamer, bei groesseren Werten schneller.
75	004B	Enthaelt $00, bei $FF haelt die Uhr an.
78	004E	Power on/off Status.
79	004F	Keine Taste gedrueckt: $18, Taste gedrueckt: $08, SHIFT und CTRL haben keine Wirkung.
80- 95	0050-005F	VON DER HAUPT-CPU BENUTZTE REGISTER:
80	0050	Register (R0H) R0: (R0H,R0L)
81	0051	Register (R0L)
82	0052	Register (R1H) R1: (R1H,R1L)
83	0053	Register (R1L)
84	0054	Register (R2H) R2: (R2H,R2L)
85	0055	Register (R2L)
86	0056	Register (R3H) R3: (R3H,R3L)
87	0057	Register (R3L)
88	0058	Register (R4H) R4: (R4H,R4L)
89	0059	Register (R4L)
90	005A	Register (R5H) R5: (R5H,R5L)
91	005B	Register (R5L)
92	005C	Register (R6H) R6: (R6H,R6L)
93	005D	Register (R6L)
94	005E	Register (R7H) R7: (R7H,R7L)
95	005F	Register (R7L)
96- 111	0060-006F	VOM MONITOR BENUTZTE REGISTER (INTERRUPT)
99- 100	0063-0064	Startadresse aus G(o)-Befehl (Monitor).
105	0069	Akkumulator A (Monitor).
106	006A	Akkumulator B (Monitor).
107	006B	Statusregister (Monitor).
108- 109	006C-006D	Indexregister (Monitor).
110- 111	006E-006F	Stack-Pointer (Stack waechst abwaerts).
112- 115	0070-0073	REGISTER FUER TASTATURROUTINE (INTERRUPT):
112	0070	Code (nicht ASCII, siehe Code-Tabelle) der zuletzt gedrueckten Taste. Bleibt solange erhalten, bis eine neue Taste gedrueckt wird. SHIFT, CTRL und GRPH haben keine Bedeutung.
116- 119	0074-0077	SERIAL INTERRUPT REGISTER (S0, S1).
120	0078	Initialized Flag 1. Bit 0: Menue, Bit 1: Clock, Bit 2: Calculator, Bits 3,4,5: nicht verwendet, Bit 6: BASIC-Application, Bit 7: BASIC (0=Request Initialize, 1=Completed).
121	0079	Options-Select. Bit 0: Slave-CPU P46, Bit 1: Slave-CPU P20, Bit 2: Main CPU P17, Bit 3: immer '0', Bits 4,5,6: nicht verwendet, Bit 7: Ausschalten des RS232C-Treibers bei BREAK (1=Aus, 0=normal). Bits 2,1,0: 000 Prom Cassette 001 Reserve 010 Nichts eingesteckt 011 Reserve

1xx Micro Cassette

122	ØØ7A	Status des seriellen Ports. Bits Ø,1: 'ØØ'=stop RS232, 'Ø1'=reading by interrupt mode, '1Ø'=reading by read one character mode, Bit 2: RS232 Ausfuehrung/Pause (Ø=Ausfuehrung, wenn Bits Ø,1 nicht Ø, 1=Pause), Bit 3: RS232-Treiber (Ø=aus, 1=ein), Bit 4: Serial-Treiber (Ø=aus, 1=ein), Bits 5,6,7: ØØØ Lesen ext.Cass. ØØ1 Lesen Mic.Cass. Ø1Ø Lesen RS232 Ø11 Nicht verwendet 1ØØ Schreiben ext.Cass. 1Ø1 Schreiben Mic.Cass. 11Ø,111 Nicht verwendet
123	ØØ7B	Run-Modus. Bits Ø,1,2,3: 'Ø'=BASIC, Bits 4,5: nicht verwendet, Bit 6: Display-Status (Ø=virt. Display, 1=phys. Display), Bit 7: RUN-Mode (1=interpreter, 2=machine lang.).
124	ØØ7C	Slave-CPU I/O-Status ('1' wenn ein). Bit Ø: Drucker, Bit 1: ext. Cass., Bit 2: int. Cass., Bit 3: RS232 ein (read), Bit 4: Tongeber, Bit 5: prom cassette power, Bit 6: Barcodeleser, Bit 7: Abbruch durch BREAK (Ø=normal, 1=Abbruch).
125	ØØ7D	Haupt-CPU I/O-Status ('1' wenn ein). Bit Ø: LCD read/write, Bit 1: Fortfuehr. Uebertragung zur Slave-CPU, Bit 2: Fortfuehr. der ser. Uebertragung, Bit 3: im Clock-Interrupt, Bit 4: power fail, Bit 5: off power switch, Bit 6: im PAUSE-Modus, Bit 7: im BREAK-Modus. Anwendungen: 'POKE125,16' fuer "Charge Battery Aborted", 'POKE125,32' fuer Sprung ins Menue, 'POKE125,64' Wartet, bis Taste gedrueckt wird, 'POKE125,128' fuer "Break"-Meldung.
126	ØØ7E	Software-Schalter #1. Bits Ø,1: Modus der externen Cassette ('Øx'=decided autom., '1Ø'=normal, '11'=reverse), Bits 2,3: Modus der internen Cassette (dto.), Bits 4: aktive Bank (Ø=bank Ø, 1=bank 1), Bit 5: zuletzt beim Menue verw. Bank (dto.), Bit 6: Bank in der der Interpreter ist (dto.), Bit 7: 'Ø'=normal, '1' PEEK und POKE in den ges. Speicherbereich. Anwendung: Enthaelt normalerweise 'ØØ', dann ist ein POKEn oberhalb des MEMSET-Bereiches nicht moeglich, durch 'POKE&H7E,&H8Ø' wird ein POKEn in den gesamten RAM-Speicher moeglich. Der Wert wird beim Einschalten zurueckgesetzt, nicht jedoch durch das Menue.
127	ØØ7F	Software-Schalter #2. Bit Ø: DIL-Schalter 1, Bit 1: DIL-Schalter 2, Bit 2: DIL-Schalter 3, Bit 3: DIL-Schalter 4, Bit 4: enable Bit Ø...3 (Ø=disable, 1=enable), Bit 5: enable Bit 7 (dto.), Bit 6: nicht verwendet, Bit 7: Drucker Ein/Aus-Schalter. Anwendung zur Zeichensatzumschaltung: $ØØ = wie durch DIL-Schalter gewaehlt (Standard), $1Ø = Schweden (ASCII), $11 = Deutschland (ASCII),

		$12 = Frankreich (ASCII), $13 = Daenemark, $14 = Schweden, $15 = Deutschland, $16 = Frankreich, $17 = Norwegen. Kann durch POKE umgestellt werden, bei einem Neustart ueber das Menue wird wieder der Standardzeichensatz eingeschaltet. Druckerschaltung: $ØØ = eingeb. Drucker normal, $AØ = Drucker immer ein, $2Ø = Drucker immer aus.
131	ØØ83	#Ø fuer DBL-Rechnungen, Bedeutung noch nicht ganz klar.
133	ØØ85	Typ des Arguments bei allen Rechnungen, insb. USR.
156- 157	ØØ9C-ØØ9D	Zeiger auf den Anfang des aktuellen Programmspeichers (auf das 1. Byte der "4-Byte-Zeilennummer").
158- 159	ØØ9E-ØØ9F	Zeiger auf den Beginn der Variablen.
160- 161	ØØAØ-ØØA1	Zeiger auf den Beginn der Arrays.
162- 163	ØØA2-ØØA3	Zeiger auf das Ende der Arrays = maximales Ende des Stacks (Stack waechst abwaerts).
164- 165	ØØA4-ØØA5	Zeiger auf das letzte Byte (niedrigere Adresse) des definierten (CLEAR) Stringbereichs (Strings wachsen abwaerts).
166- 167	ØØA6-ØØA7	Zeiger auf das aktuelle Ende der Strings (wachsen abwaerts).
170- 171	ØØAA-ØØAB	Zeiger auf das erste Byte (hoehere Adresse) des definierten (CLEAR) Stringbereichs (Strings wachsen abwaerts).
172- 173	ØØAC-ØØAD	Zeiger auf die aktuelle Programm-Zeilennummer (wird bei Direktkommandos auf "FF FF" gesetzt).
174- 175	ØØAE-ØØAF	Zeiger auf die aktuelle oder zuletzt abgearbeitete (vor END, STOP, BREAK) Programm-Zeilennummer (letzte Zeilennummer bleibt erhalten).
176- 177	ØØBØ-ØØB1	Niedrigste bisher abgearbeitete Programm-Zeilennummer. Nach Start auch Anzahl der Programmbytes.
178- 179	ØØB2-ØØB3	Aktuelle Speicherstelle des naechsten BASIC-Befehls. Dient nach STOP/BREAK/END fuer CONT. Wird auf "ØØ ØØ" gesetzt, wenn Fortfuehrung nicht moeglich, dann 'CN Error'.
180- 181	ØØB4-ØØB5	Aktuelle Speicherstelle des naechsten BASIC-Befehls (bleibt erhalten). Zeigt bei Direktkommandos in den Eingabepuffer (ab $Ø756).
184- 185	ØØB8-ØØB9	Zeiger auf den Anfang des aktuellen Programmspeichers (auf das "ØØ-Byte" vor der ersten "4-Byte-Zeilennummer").
213- 221	ØØD5-ØØDD	Fliesspunkt-Akku (Byte 1-9). Hier wird z.B. das Argument fuer "USR" uebergeben. Das Format: INTEGERS: Byte 1 und 2 immer $ØØ, Byte 3 ist das HIGH BYTE und Byte 4 das LOW BYTE der vorzeichenbehafteten Integer-Zahl (-32768 = $8ØØØ ... +32767 = $7FFF), Bytes 5-9 haben keine Bedeutung.

EINFACHE GENAUIGKEIT: Byte 1 enthaelt den Exponenten mit einem Bias von $8Ø, d.h.: Exponent $81 bedeutet Rechtsshift um 1, Exponent $7F bedeutet Linksshift um 1 Byte. Bytes 2-4 enthalten die Mantisse, das Vorzeichenbit in Byte 2 ist immer gesetzt. Die Mantisse selber wird nach links geshiftet, bis das hoechstwertige Bit aus der Zahl herausgeschoben wurde (Normalisierung). Byte 5-8 haben keine Bedeutung, Byte 9 ist das Vorzeichen (+ = $ØØ / - = $FF).

DOPPELTE GENAUIGKEIT: Hier werden die Bytes 5-8 mit benutzt, sonst wie bei einfacher Genauigkeit.

STRINGS: Im Byte 3 und 4 erscheint der Wert, der auch bei der BASIC-Funktion VARPTR angegeben wuerde.

BEISPIELE

4	xx xx ØØ Ø4 xx xx xx xx ØØ
4.Ø	83 8Ø ØØ ØØ xx xx xx xx ØØ
.25	7F 8Ø ØØ ØØ xx xx xx xx ØØ
-4	xx xx FF FC xx xx xx xx ØØ
-4.Ø	83 8Ø ØØ ØØ xx xx xx xx FF
- .25	7F 8Ø ØØ ØØ xx xx xx xx FF
1234567891234567	B3 8C 5A A7 93 75 EØ EØ ØØ
strings (Beispiel)	xx xx 4A FD xx xx xx xx xx

Bei USR-Aufruf wird das Argument in den Fliesskomma-Akku uebernommen (ausserdem enthaelt Adresse 2Ø1=$ØØC9 "$FF" bei einer Zahl mit Nachkommastellen und 2ØØ=$ØØC8 die Anzahl der Nachkommastellen). Bei der Rueckkehr durch 'RTS' wird der Inhalt des Akkus an die Funktion uebergeben. Anzumerken ist noch, dass beim USR-Aufruf das Indexregister mit $ØØD5 (numerisch) oder dem VARPTR (Strings) geladen wird.

227- 235	ØØE3-ØØEB	zweiter Fliesspunktakku entsprechend $D5-DD.
25Ø	ØØFA	$B6 'LDAA,ext.'
251- 252	ØØFB-ØØFC	Zeiger in laufendes BASIC-Programm oder Eingabepuffer (bei Direktkommandos).
253- 255	ØØFD-ØØFF	JMP $B3D8 (Abschluss Zeichenholroutine)
256- 258	Ø1ØØ-Ø1Ø2	JMP $EF25 (Clock Interrupt Jump)
259- 261	Ø1Ø3-Ø1Ø5	JMP $EF24 (Ext. Port Interrupt IRQ1)
262- 264	Ø1Ø6-Ø1Ø8	JMP $DFFA (Trap)
265- 267	Ø1Ø9-Ø1ØB	JMP $EDD7 (SCI, RDRF+ORFE+TDRE)
268- 27Ø	Ø1ØC-Ø1ØE	JMP $EF2C (TOF, Timer Overflow)
271- 273	Ø1ØF-Ø111	JMP $F51Ø (OCF, Timer Output Compare)
274- 276	Ø112-Ø114	JMP $EF24 (ICF, Timer Input Capture)
277- 279	Ø115-Ø117	JMP $EED6 (IRQ-Vektor)
28Ø- 282	Ø118-Ø11A	Hier kann man den SWI-Vektor definieren als JMP-Befehl: "7E xx xx" in 3 Bytes. Normal enthalten die Bytes Ø.
283- 285	Ø11B-Ø11D	Hier kann man den NMI-Vektor definieren als JMP-Befehl: "7E xx xx" in 3 Bytes. Normal

		enthalten die Bytes Ø.
286- 287	Ø11E-Ø11F	Zeigt auf den Bereich (Feld mit 192 Bytes) in dem eigene Zeichen definiert werden koennen. Siehe Kapitel DEFINITION EIGENER ZEICHEN.
288- 289	Ø12Ø-Ø121	Jump-Vektor wenn BREAK gedrueckt wird (nicht in BASIC).
29Ø- 291	Ø122-Ø123	Jump-Vektor wenn MENU gedrueckt wird (nicht in BASIC).
292- 293	Ø124-Ø125	Jump-Vektor wenn PAUSE gedrueckt wird (nicht in BASIC).
294- 295	Ø126-Ø127	Jump-Vektor wenn CTRL-PF3 gedrueckt wird.
296- 297	Ø128-Ø129	Jump-Vektor wenn CTRL-PF4 gedrueckt wird.
298- 299	Ø12A-Ø12B	Jump-Vektor wenn CTRL-PF5 gedrueckt wird.
3ØØ- 3Ø1	Ø12C-Ø12D	Hoechste RAM-Adresse+1.
3Ø2- 3Ø3	Ø12E-Ø12F	Anzahl Bytes im lfd. ROM-File.
3Ø4- 3Ø5	Ø13Ø-Ø131	vector power on by wake up, called procedure address
3Ø6- 3Ø7	Ø132-Ø133	vector power off by wake up, called procedure address
3Ø8- 3Ø9	Ø134-Ø135	Zeiger auf Ende der BASIC-Programme.
31Ø- 311	Ø136-Ø137	Zeiger auf Beginn des BASIC-Speicherbereichs (normal "ØA 4Ø"), kann durch MEMSET veraendert werden, um bei $ØA4Ø ein Maschinenprogramm beginnen zu lassen. Diese Adresse wird z.B. fuer STAT benoetigt.
312- 313	Ø138-Ø139	Zeiger auf Einsprung zur 'garbage collection'.
314- 315	Ø13A-Ø13B	header by menu: exist header flag in each rom.
316- 319	Ø13C-Ø13F	heading by user's program (used by menu).
32Ø- 399	Ø14Ø-Ø18F	TASTATUR-ROUTINEN-ARBEITSBEREICH:
32Ø	Ø14Ø	Laenge des Tastaturpuffers (Ø...8,Standard8).
321	Ø141	Auto-Repeat-Intervall (1).
322	Ø142	Auto-Repeat-Intervall (2).
323- 324	Ø143-Ø144	sampling time.
325- 344	Ø145-Ø157	Zur Zeit gedrueckte Taste:
325&335	Ø145&Ø14F	Ø=1, 1=2, 2=4, 3=8, 4=16, 5=32, 6=64, 7=128
326&336	Ø146&Ø15Ø	8=1, 9=2, ae=4, oe=8, ,=16, sz=32, .=64, -=128
327&337	Ø147&Ø151	ue=1, A=2, B=4, C=8, D=16, E=32, F=64, G=128
328&338	Ø148&Ø152	H=1, I=2, J=4, K=8, L=16, M=32, N=64, O=128
329&339	Ø149&Ø153	P=1, Q=2, R=4, S=8, T=16, U=32, V=64, W=128
33Ø&34Ø	Ø14A&Ø154	X=1, Z=2, Y=4, +=8, #=16, Hochpfeil=32, ->=64, <-=128
331&341	Ø14B&Ø155	RET=1, SPC=2, TAB=4, <=32, GRPH=64, CAPS=128
332&342	Ø14C&Ø156	CLR=1, SCRN=2, DEL=16, MENU=32
333&343	Ø14D&Ø157	PF1=1, PF2=2, PF3=4, PF4=8, PF5=16, P.Feed=32
334&344	Ø14E&Ø158	Zustand des DIL-Schalters, Druckerschalters, SHIFT und CTRL. Bit Ø=Schalter 4, Bit 1=Schalter 3, Bit 2=Schalter 2, Bit 3=Schalter 1, Bit 5=SHIFT, Bit 6=CTRL (jeweils EIN=gesetzt, AUS=nicht gesetzt), Bit 7=Druckerschalter. Aenderung erfolgt erst, nachdem andere Taste gedrueckt wurde.
345- 354	Ø159-Ø162	TABELLE ZUR ERMITTLUNG DER GEDRUECKTEN TASTE:
351	Ø15F	Normal 'BF', kurze Pulse nach loslassen der Taste: "RETURN"='BE' "CAPS"='3F' "TAB"='BB' "<"='9F'.

352	Ø160	Normal 'FF', kurze Pulse nach loslassen der Taste: "MENU"='DF' "CLR"='FE' "SCRN"='FD' "INS"='EF'.
353	Ø161	Normal 'FF', kurze Pulse nach loslassen der Taste: "PF1"='FE "PF2"='FD' "PF3"='FB' "PF4"='F7' "PF5"='EF' "PAPER FEED"='DF'.
354	Ø162	Normal 'FF', kurze Pulse nach loslassen der Taste: Irgendeine Taste nach "CTRL"-Taste 'BF', nach "SHIFT"-Taste 'DF'.
355- 356	Ø163-Ø164	Zeiger auf den Anfang des Speicherbereichs fuer die K(ey)-Funktion (initial stack).
357	Ø165	'ØA' wenn gueltige Bytes von K(ey)-Funktion vorhanden, 'ØØ', wenn K(ey)-Funktion geloescht wurde, 'Ø1', wenn beim Einschalten keine gueltigen K(ey)-Werte angetroffen wurden, 'ØB', wenn K-Funktion in Arbeit.
358	Ø166	Anzahl Bytes in K(ey)-Puffer, nur gueltig, wenn in 357 ($Ø165) 'ØA' steht, wird beim Loeschen des Puffers nicht zurueckgesetzt.
359	Ø167	Zaehler, fuer K(ey)-Funktion.
36Ø	Ø168	Anzahl Bytes im Tastaturpuffer (Ø...8).
361	Ø169	Flag zu der zuletzt gedrueckten Taste (key mode): ohne SHIFT=$ØØ, CAPS=$X4, GRPH=$1X, SHIFT=$2X, CTRL=$4X, CTRL+SHIFT=$6X, GRPH+SHIFT=$BX. Erlaubt programmmaessige Umschaltung: 'POKE&H169,2' schaltet auf NUM LOCK, 'POKE&H169,4' schaltet auf CAPS LOCK, 'POKE&H169,Ø' ist der Normalzustand.
362	Ø16A	Keine Taste gedrueckt: $FF, CAPS gedrueckt: $ØØ, andere Taste gedrueckt: $Ø1. GRPH, SHIFT und CTRL haben keine Wirkung.
363	Ø16B	Zaehler fuer Tastatur-REPEAT.
364	Ø16C	Wie 112 ($ØØ7Ø). Fuer REPEAT-Funktion.
365	Ø16D	ASCII-Code der zuletzt gedrueckten Taste (siehe ASCII-Tabelle). Funktionstasten ergeben "$FE".
366	Ø16E	Zuletzt gedrueckte Funktionstaste: F1=$F1 ... F1Ø=$FA.
367- 384	Ø16F-Ø18Ø	18-Bytes-Speicher fuer 'K(ey)'-Funktion des Monitors.
385- 392	Ø181-Ø188	Tastaturpuffer fuer 8 Zeichen. Abspeicherung im ASCII-Code.
4ØØ- 43Ø	Ø19Ø-Ø1AE	MICRO-DRUCKER-ARBEITSBEREICH:
4Ø6	Ø196	Anzahl Daten im Druckerpuffer.
4Ø7- 43Ø	Ø197-Ø1AE	24 Bytes-Puffer fuer eine Druckzeile. Abspeicherung im ASCII-Code. In den Puffer kann man schreiben und den Inhalt dann mit "LPRINT" bzw. "JSR $ABØØ" ausdrucken.
431- 451	Ø1AF-Ø1C3	RS232C-ARBEITSBEREICH:
431- 432	Ø1AF-Ø1BØ	Definierte Baud-Rate aus 'COMØ:'-Definition: 'Ø'=$15D1, '1'=$1ØØØ, '2'=$Ø8ØØ, '3'=$Ø4ØØ, '4'=$Ø2ØØ, '5'=$Ø1ØØ, '6'=$ØØ8Ø.
433- 434	Ø1B1-Ø1B2	generating polynomial.
435- 436	Ø1B3-Ø1B4	BCC-Register.
437	Ø1B5	Definierte Wortlaenge aus 'COMØ:'-Definition: $Ø8 = 8 Bit, $Ø7 = 7 Bit.

438	Ø1B6	Definierte Paritaet, Stopbits, Steuerleitungen aus 'COMØ:'-Definition: Bit Ø+1 = Stopbits ('Ø1' ein, '1Ø' zwei Stopbits), Bit 2-5 = Steuerleitungen (1...F), Bit 6+7 = Paritaet ('ØØ' even, 'Ø1' odd, '1Ø' non).
439	Ø1B7	Fehlerstatus.
44Ø- 441	Ø1B8-Ø1B9	Puffer-Anfangsadresse der empfangenen Daten.
442- 443	Ø1BA-Ø1BB	Puffer-Endadresse+1 der empfangenen Daten.
444- 445	Ø1BC-Ø1BD	Eingangspufferlaenge.
446- 447	Ø1BE-Ø1BF	Zeiger zum Schreiben in den Puffer.
448- 449	Ø1CØ-Ø1C1	Zeiger zum Lesen aus dem Puffer.
45Ø- 451	Ø1C2-Ø1C3	Anzahl Daten im Puffer.
452- 468	Ø1C4-Ø1CF	SERIAL-COMMUNICATION-ARBEITSBEREICH:
452	Ø1C4	'FMT'-Wert.
453	Ø1C5	'DID'-Wert.
454	Ø1C6	'SID'-Wert.
455	Ø1C7	'FNC'-Wert.
456	Ø1C8	'SIZ'-Wert.
457	Ø1C9	Empfangener Charakter fuer 'ACK'.
458	Ø1CA	transmittet try counter.
459	Ø1CB	received character time out time.
46Ø	Ø1CC	received try limit.
461	Ø1CD	received 'ack' time out time.
462	Ø1CE	Slave/Master-Modus.
463	Ø1CF	idling time after sending 'eot'.
464- 468	Ø1DØ-Ø1D4	nicht verwendet.
469- 491	Ø1D5-Ø1EB	EXT. CASSETTENRECORDER-ARBEITSBEREICH:
469	Ø1D5	current mode.
47Ø- 471	Ø1D6-Ø1D7	current block number.
472- 473	Ø1D8-Ø1D9	value of bcc register (slave cpu).
474- 475	Ø1DA-Ø1DB	nicht verwendet.
476	Ø1DC	block gap mode.
477	Ø1DD	error flag.
478- 479	Ø1DE-Ø1DF	Puffer-Anfangsadresse.
48Ø- 481	Ø1EØ-Ø1E1	Puffer-Endadresse +1.
482- 483	Ø1E2-Ø1E3	Pufferlaenge (Bytes).
484- 485	Ø1E4-Ø1E5	Zeiger zum Schreiben in den Puffer.
486- 487	Ø1E6-Ø1E7	Zeiger zum Lesen aus dem Puffer.
488- 489	Ø1E8-Ø1E9	Anzahl Daten im Puffer.
49Ø	Ø1EA	read try limit count.
491	Ø1EB	read tried count
492- 519	Ø1EC-Ø2Ø7	MICRO-CASSETTENRECORDER-ARBEITSBEREICH:
492	Ø1EC	current mode.
493- 494	Ø1ED-Ø1EE	current block number.
495- 496	Ø1EF-Ø1FØ	value of bcc register (slave cpu).
497- 498	Ø1F1-Ø1F2	nicht verwendet.
499	Ø1F3	block gap mode.
5ØØ	Ø1F4	error flag.
5Ø1- 5Ø2	Ø1F5-Ø1F6	Puffer-Anfangsadresse.
5Ø3- 5Ø4	Ø1F7-Ø1F8	Puffer-Endadresse +1.
5Ø5- 5Ø6	Ø1F9-Ø1FA	Puffer-Laenge (Bytes).
5Ø7- 5Ø8	Ø1FB-Ø1FC	Zeiger zum Schreiben in den Puffer.
5Ø9- 51Ø	Ø1FD-Ø1FE	Zeiger zum Lesen aus dem Puffer.
511- 512	Ø1FF-Ø2ØØ	Anzahl Daten im Puffer.

513	Ø2Ø1	read try limit count.
514	Ø2Ø2	read tried count.
515- 516	Ø2Ø3-Ø2Ø4	Wert von TAPCNT (Bandzaehlwerk).
517	Ø2Ø5	Aktuelles Kommando bei Handbetrieb.
518-519	Ø2Ø6-Ø2Ø7	Arbeitsbereich fuer das Bandzaehlwerk.
52Ø- 527	Ø2Ø8-Ø2ØF	ROM-CARDRIDGE-ARBEITSBEREICH:
52Ø	Ø2Ø8	Status
521- 522	Ø2Ø9-Ø2ØA	Adresszaehler
523- 524	Ø2ØB-Ø2ØC	Anfangsadresse des Files.
525- 526	Ø2ØD-Ø2ØE	Endadresse +1 des Files.
527- 538	Ø2ØF-Ø21A	MONITOR READ/WRITE-ARBEITSBEREICH:
527- 528	Ø2ØF-Ø21Ø	Adresse "T" des aktuellen Maschinenprogramms.
529- 53Ø	Ø211-Ø212	Adresse "L" des aktuellen Maschinenprogramms.
531- 532	Ø213-Ø214	Adresse "O" des aktuellen Maschinenprogramms.
533- 534	Ø215-Ø216	Adresse "E" des aktuellen Maschinenprogramms.
535	Ø217	Geraet.
536	Ø218	Status (Dummy).
537- 538	Ø219-Ø21A	R/W-Routine (entry point).
539	Ø21B	fuer Micro-Cassette (Bandzaehlwerk).
54Ø- 543	Ø21C-Ø21F	nicht benutzt.
544- 671	Ø22Ø-Ø29F	DISPLAY-ARBEITSBEREICH:
544- 623	Ø22Ø-Ø26F	Speicherbereich fuer den Inhalt des aktuellen (physikalischen) Bildschirms (2Øx4 Zeichen). Kann nicht ueberschrieben (POKE, STA..), aber gelesen (PEEK, LDA..) werden.
624- 625	Ø27Ø-Ø271	Zeiger auf den Anfang des virtuellen Bildschirmspeichers (erstes Byte der ersten Zeile).
626- 627	Ø272-Ø273	Zeiger auf das Ende des virtuellen Bildschirmspeichers (letztes Byte der letzten Zeile).
628- 629	Ø274-Ø275	'Home'-Position des physik. Schirms im Puffer.
63Ø	Ø276	Breite des virt. Schirms -1 (X).
631	Ø277	Laenge des virt. Schirms -1 (Y).
632	Ø278	X-Pos. des Cursors im physik. Schirm (Ø-19).
633	Ø279	Y-Pos. des Cursors im physik. Schirm (Ø-3).
634	Ø27A	Zaehler fuer Links/Rechts-Scroll.
635	Ø27B	Zaehler fuer Hoch/Runter-Scroll.
636	Ø27C	Cursor-Rand.
637	Ø27D	Scroll-Geschwindigkeit (Ø-9).
638	Ø27E	Zeiger, wo Charakter angezeigt wird (X: Ø-19).
639	Ø27F	Zeiger, wo Charakter angezeigt wird (Y: Ø-3).
64Ø	Ø28Ø	Display-Status. Bit 7: refresh screen (1 = ganzen Schirm, Ø = nur ein Zeichen), Bit 6: Cursor-Flag (1 = Cursor ein, Ø = aus), Bit 5: Cursor-Schalter um einen Char. zu schreiben (1 = Cursor (Unterstreichung) ein, Ø = aus), Bit 4: scroll delay flag (1 = warte vor naechstem Zeichen, Ø = warte nicht), Bit 3,2,1: nicht verwendet, Bit Ø: List-Flag (1 = festes Fenster, Ø = nicht fest).
641- 645	Ø281-Ø285	In neueren Versionen nicht verwendet.
646- 651	Ø286-Ø28B	6 Bytes Charakter-Zwischenspeicher.

672- 719	Ø2AØ-Ø2CF	MONITOR-ARBEITSBEREICH:
672- 673	Ø2AØ-Ø2A1	Break-Point-Adresse.
674	Ø2A2	Byte auf Break-Point.
675	Ø2A3	Zwischenspeicherung des LCD-Status (von $Ø28Ø).
676	Ø2A4	R/W-PACKET (Modus)
677- 678	Ø2A5-Ø2A6	(Puffer-Adresse)
679- 686	Ø2A7-Ø2AE	(File-Name)
687- 694	Ø2AF-Ø2B6	(File-Typ)
695- 696	Ø2B7-Ø2B8	(Top Address)
697- 698	Ø2B9-Ø2BA	(Last Address)
699- 7ØØ	Ø2BB-Ø2BC	(Ofset Value)
7Ø1- 7Ø2	Ø2BD-Ø2BE	(Entry Point)
7Ø3- 7Ø4	Ø2BF-Ø2CØ	Program-Counter.
7Ø5- 7Ø6	Ø2C1-Ø2C2	Startwert des Program-Counters beim Einsprung in den Monitor ($A3B5 vom BASIC, $D23B vom Menue) = Ruecksprungadresse aus Monitor.
7Ø7- 7Ø8	Ø2C3-Ø2C4	Adresse: Letzte Spalte der ersten Zeile des LCD (Puffer).
7Ø9	Ø2C5	Zwischenspeicherung des 'Run-Modus' (aus $ØØ7B).
72Ø- 879	Ø2DØ-Ø36F	SPEICHERBEREICH FUER DAS MENUE. Beim Aufruf des Menues zeigt der Pointer fuer den Bildschirmspeicher direkt auf diesen Bereich. Das Menue wird hierbei immer aktualisiert. Abschluss durch Ø-Bytes. Dieser Bereich wird auch fuer den Cassettenbetrieb mitverwendet, wird beim Menue aber evtl. ueberschrieben:
72Ø- 779	Ø2DØ-Ø3Ø6	'CTRL/Paragraph Initialize 1.Montor 2.Basic'
78Ø- 799	Ø3Ø7-Ø31F	'3.'Programmname 1.
8ØØ- 819	Ø32Ø-Ø333	'4.'Programmname 2.
82Ø- 839	Ø334-Ø347	'5.'Programmname 3.
84Ø- 859	Ø348-Ø35B	'6.'Programmname 4.
86Ø- 879	Ø35C-Ø36F	'7.'Programmname 5.
72Ø- 8Ø3	Ø2DØ-Ø323	HEADER EXTERNER RECORDER.
72Ø	Ø2DØ	Block Markierung 'H'.
721- 722	Ø2D1-Ø2D2	Block Nummer.
723	Ø2D3	Blocklaenge.
724- 727	Ø2D4-Ø2D7	'HDR1'.
728- 735	Ø2D8-Ø2DF	File-Name.
736- 743	Ø2EØ-Ø2E7	File-Typ.
739	Ø2E3	Dateiart (Ø-BASIC, 1-Datei, 2-Objekt).
744	Ø2E8	Record-Typ.
745	Ø2E9	block gap mode.
746- 75Ø	Ø2EA-Ø2EE	Block-Laenge.
751- 755	Ø2EF-Ø2F3	reserviert.
756- 761	Ø2F4-Ø2F9	Datum.
762- 767	Ø2FA-Ø2FF	Zeit.
768- 773	Ø3ØØ-Ø3Ø5	reserviert.
774- 775	Ø3Ø6-Ø3Ø7	volume number.
776- 783	Ø3Ø8-Ø3ØF	System-Name 'HX-2Ø ...'.
784- 8Ø3	Ø31Ø-Ø323	reserviert.
8Ø4- 887	Ø324-Ø377	HEADER MICROCASSETTENRECORDER:
8Ø4	Ø324	Block-Markierung 'H'.
8Ø5- 8Ø6	Ø325-Ø326	Block-Nummer.

807	0327	Block-Laenge.
808- 811	0328-032B	'HDR1'.
812- 819	032C-0333	File-Name.
820- 827	0334-033B	File-Typ.
823	0337	Dateiart (0-BASIC, 1-Datei, 2-Objekt).
828	033C	Record-Typ.
829	033D	block gap mode.
830- 834	033E-0342	Block-Laenge.
835- 839	0343-0347	reserviert.
840- 845	0348-034D	Datum.
846- 851	034E-0353	Zeit.
852- 857	0354-0359	reserviert.
858- 859	035A-035B	volume number.
860- 867	035C-0363	System-Name 'HX-20...'.
868- 887	0364-0377	reserviert.
888-1147	0378-047B	I/O-Puffer fuer LOAD, SAVE, Monitor. 260 Bytes (default).
1149-1999	047D-04AF	Stack (default).
1202-1203	04B2-04B3	Obergrenze des RAM-Speichers +1. $4000 bei 16K-, $8000 bei 32K-RAM.
1204	04B4	Nummer des aktuellen BASIC-Speicherbereichs (1...5).
1205-1206	04B5-04B6	Zeiger auf Beginn des aktuellen BASIC-Speicherbereichs (auf das erste Byte des "Headers"). Diese Adresse wird z.B. fuer LIST und SAVE verwendet.
1270-1271	04F6-04F7	Relative Adressen RAM-File.
1272	04F8	Recordlaenge RAM-File.
1278-1279	04FE-04FF	Laenge RAM-File.
1280-1281	0500-0501	Zeiger auf den Beginn des gesamten BASIC-Speicherbereichs ($0A40 = 1.Byte des "Headers" von P1, wenn nicht mit MEMSET umdefiniert).
1282-1439	0502-059F	Belegung der Funktionstasten (Standard, oder wie definiert).
1442-1443	05A2-05A3	Zeiger auf Beginn des definierten (CLEAR) RAM-File-Speichers.
1506-1636	05E2-0664	JMP-TABELLEN UND ROUTINEN:
1617-1619	0651-0653	'JMP 88DF' Hierher springt das Betriebssystem bei Aufruf des WHILE-Befehls. Da dieser Befehl nicht implementiert ist, kann er fuer eigene Zwecke benutzt werden, wenn dieser JMP-Befehl entsprechend geaendert wird.
1620-1622	0654-0656	'JMP 88DF' Hierher springt das Betriebssystem beim Aufruf des WEND-Befehls.
1679-1683	068F-0693	COM0-Spezifikation in ASCII nach deren Definition.
1692-1872	069C-0750	LISTE DER PERIPHERIEGERAETE (incl. Vektoren und Parameter):
1740	06CC	Virtuelle Displaybreite (normal 40 = $28).
1766	06E6	Druckbreite RS-232C (normal 80 = $50).
1870	074E	Druckbreite eingeb. Drucker (normal 24 = $18).
1876-1877	0754-0755	Nummer der zuletzt eingegebenen Zeile.
1878-2133	0756-0855	Eingabepuffer (255 Bytes) fuer

		Programmeingabe, Direktkommandos, INPUT-Befehl (hier wird die Eingabe ab 1879=$Ø757 als ASCII-Code gespeichert - 1. Byte bei INPUT $2C, bei LINEINPUT $97), Abschluss durch ØØ-Byte.
2140-2155	Ø85C-Ø86B	16-Byte-Puffer fuer aktuellen Variablennamen.
2190	Ø88E	Anzahl der OPTION BASE-Aufrufe.
2191	Ø88F	OPTION BASE (Ø oder 1).
2214	Ø8A6	$ØØ bei TROFF, $4F bei TRON.
2217	Ø8A9	Letzter Fehler-Code fuer "ERR"-Befehl.
2218-2219	Ø8AA-Ø8AB	Letzte Fehlerzeile fuer "ERL"-Befehl.
2229-2254	Ø8B5-Ø8CE	DEF-Typ der Variablennamen ($8B5 fuer 'A' - $8CE fuer 'Z').
2255-2274	Ø8CF-Ø8E2	USR-VEKTOREN-TABELLE. Wenn nichts definiert wurde, zeigen alle Vektoren auf &8C7Ø. Naeheres zum USR-Befehl bei Adresse 213-221.
2255-2256	Ø8CF-Ø8DØ	Vektor fuer USRØ.
2257-2258	Ø8D1-Ø8D2	Vektor fuer USR1.
2259-226Ø	Ø8D3-Ø8D4	Vektor fuer USR2.
2261-2262	Ø8D5-Ø8D6	Vektor fuer USR3.
2263-2264	Ø8D7-Ø8D8	Vektor fuer USR4.
2265-2266	Ø8D9-Ø8DA	Vektor fuer USR5.
2267-2268	Ø8DB-Ø8DC	Vektor fuer USR6.
2269-227Ø	Ø8DD-Ø8DE	Vektor fuer USR7.
2271-2272	Ø8DF-Ø8EØ	Vektor fuer USR8.
2273-2274	Ø8E1-Ø8E2	Vektor fuer USR9.
2283-2293	Ø8EB-Ø8F5	Dateiname der letzten Eingabe fuer SAVE/LOAD/OPEN als ASCII-Code (erstes Zeichen fehlt).
2299-23Ø9	Ø8FB-Ø9Ø5	Gefundener Dateiname bei BASIC-LOAD/OPEN von Cassettenfiles als ASCII-Zeichen. Den Namen des zuletzt geladenen Files bekommt man durch: FORI=2299TO23Ø9:PRINTCHR$(PEEK(I));:NEXT
2624	ØA4Ø	Beginn des BASIC-Programmspeicherbereichs, wenn er durch MEMSET nicht umdefiniert wurde.
2624	ØA4Ø	Beginn eines Maschinenprogramms, das Ende+1 muss mit MEMSET festgelegt werden!

Erläuterung: Die erste Spalte enthält immer die dezimale Form der Adresse, die zweite Spalte die hexadezimale Form.

Die Zeiger (Pointer) werden in 2 Bytes gespeichert, das erste enthält das HIGY BYTE und das zweite das LOW BYTE der Adresse.

Folgendermassen bekommt man (vom BASIC aus) die dezimale Adresse:

```
? PEEK(High Byte Adresse)*256+PEEK(Low Byte Adresse)
```

ROM-ROUTINEN

Die Folgende Tabelle enthaelt ROM-Routinen, welche von BASIC- oder Maschinenspracheprogrammen direkt aufgerufen werden koennen. Wenn Parameter in CPU-Registern uebergeben werden, dann nur in Maschinenspracheprogrammen . Die Einsprungadressen sind jeweils dezimal und hexadezimal angegeben. Am Ende des EPSON-ROM-Speichers steht eine umfangreiche JMP-Tabelle, die (wenn moeglich) fuer die Einsprungadressen verwendet wird.

Routinen, welche mit symbolischen Namen bezeichnet sind, werden spaeter ausfuehrlich erlaeutert.

32768	8ØØØ	BEGINN DES ROM-SPEICHERS (Standard)
32785	8Ø11	(bis $8ØBØ) Tabelle: Sprungvektoren fuer BASIC-Befehle.
32945	8ØB1	(bis $83Ø3) Tabelle: BASIC-Befehle in ASCII.
33541	83Ø5	(bis $8358) Tabelle: Sprungvektoren fuer BASIC-Befehle.
33843	8433	Ausgabe einer Fehlermeldung: Fehlercode in Akku B, dann 'JMP 8433' zur Ausgabe der Fehlermeldung. 'ØØ' in Akku B gibt nur die Meldung ' Error'.
34576	871Ø	Meldung "UL Error".
34818	88Ø2	Meldung "OS Error".
35Ø39	88DF	Meldung "SN Error".
35Ø74	89Ø2	Meldung "MO Error".
35952	8C7Ø	Meldung "FC Error".
38Ø39	9497	Fuehrt "NEW" aus, auch wenn Programm mit TITLE versehen ist! (DELETE alles)
38113	94E1	Fuehrt "AUTO" aus.
4ØØ88	9C98	Fuehrt "RENUM" aus.
419Ø9	A3B5	BASIC-Warmstart.
426Ø9	A671	Meldung "DU Error".
42663	A6A7	Fuehrt "MERGE" aus.
42671	A6AF	Fuehrt "LOAD" aus.
43761	AAF1	Fuehrt "LLIST" aus.
43773	ABØØ	Fuehrt "LPRINT" aus. Um eine Zeile Text auszudrucken, den Text (ASCII-Code) in den Druckerpuffer (Adressen 4Ø7-43Ø = $Ø197-Ø1AE) schreiben und LPRINT aufrufen.
44134	AC66	Fuehrt "WIND" aus.
44289	ADØ1	Fuehrt "LOADM" aus.

```
44420  AD84  Fuehrt "COPY" aus.

44560  AE10  Fuehrt "LOAD?" aus.

44893  AF5D  Fuehrt "FILES" aus.

44986  AFBA  (bis $AFD4) Tabelle: "Searching Found: Skip" in ASCII.

45615  B22F  Sprung in P1 mit Copyright-Meldung.

46059  B3EB  (bis $B424) Tabelle: "EPSON BASIC  Copyright"-Meldung  in
             ASCII.

46117  B425  (bis $B4DA) Tabelle: Liste     und     Parameter      der
             Peripheriegeraete, wird beim Menue ins  RAM  kopiert  (ab
             $069C).

             ARITHMETIKROUTINEN: D = Akku $D5-DD, E= Akku $E3-EB

49390  C0F1  D = D + 0.5
49398  C0F6  Lade E und D = E - D  [2]
49401  C0F9  D = E - D             [1]
49409  C101  Lade E und D = E + D  [2]
49412  C104  D = E + D             [1]
49924  C304  Lade E und D = E * D  [2]
49929  C309  D = E * D             [1]
50086  C3A6  Lade E                [2]
50172  C3FC  D = 10 * D    Nur fuer SNG
50209  C421  D = D / 10    Nur fuer SNG
50212  C424  D = E / 10    Nur fuer SNG
50223  C42F  Lade E und D = E / D  [2]
50226  C432  D = E / D             [1]
50435  C503  Lade D                [2]
50497  C541  Speichere D           [2]
50550  C576  E nach D
50581  C595  D nach E
50679  C5F7  Vergleiche D und E
50690  C602  Vergleiche D und Var. [2]
             Ergebnis: Reg.B = 0 fuer D=
                       Reg.B = 1 fuer D>
                       Reg.B =FF fuer D<
51202  C804  Ausgabe des kombinierten Inhalts der beiden Akkus.
52483  CD03  D = -D

             ANMERKUNGEN: [1] Reg.B muss hier Inhalt von D5 enthalten
                          [2]  Reg.X   muss   VARPTR   der   Variablen
             enthalten

47173  B845  Fuehrt "CLS" aus, loescht Display.

48740  BE64  (bis $BEAA) Tabelle: Standardbelegung der Funktionstasten
             (in ASCII), wird bei der Initialisierung ins RAM  kopiert
             (ab $0502).

53262  D00E  Sprung ins Menue.

55084  D72C  (bis $D74E) Tabelle: "Trap! OK Error Protected Break"  in
             ASCII.
```

```
57342  DFFE  (bis $DFFF) Sprungvektor: Zeigt auf die Startadresse  der
             Menue-Routine ($D00E).

57344  E000  "Soft-RESET", als ob die RESET-Taste gedrueckt wuerde.

57892  E224  (bis $E24B) Tabelle: "Enter DATE and  TIME  MMDDYYHHMMSS"
             in ASCII.

61368  EFB8  (bis $EFC7) Tabelle: "CHARGE BATTERY!" in ASCII.

             JMP-TABELLE:

65233  FED1  [HSTRIO] Warmstart-Initialisierung.
65236  FED4  [WRTP26] write address 26.
65239  FED7  [BILOAD] Cassette, PROM-Cardridge, Disk: READ und CLOSE.
65242  FEDA  [OPNLOD] Cassette, PROM-Cardridge, Disk: OPEN fuer LOAD.
65245  FEDD  [BIDUMP] Cassette, Diskette: WRITE (dump) und CLOSE.
65248  FEE0  [OPNDMP] Cassette, Diskette: OPEN fuer WRITE (dump).
65251  FEE3  [DIRPRM] PROM-Cardridge: Directory lesen.
65254  FEE6  [CLSPRM] PROM-Cardridge: CLOSE.
65257  FEE9  [REDPRM] PROM-Cardridge: Einen Charakter lesen.
65260  FEEC  [OPNPRM] PROM-Cardridge: OPEN zum lesen.
65263  FEEF  [CNTMCS] Micro-Cassette: Zaehlwerk lesen/schreiben.
65266  FEF2  [SEKMCS] Micro-Cassette: Nach Zaehlwerkstand spulen.
65269  FEF5  [REWMCS] Micro-Cassette: Spulen zum Bandanfang.
65272  FEF8  [SETCLK] wurde gestrichen.
65275  FEFB  [CLSMCS] Micro-Cassette: CLOSE.
65278  FEFE  [WRITMS] Micro-Cassette: Einen Charakter schreiben.
65281  FF01  [OPNWMS] Micro-Cassette: OPEN zum schreiben.
65284  FF04  [READMS] Micro-Cassette: Einen Charakter lesen.
65287  FF07  [SRCRMS] Micro-Cassette: File suchen und OPEN zum lesen.
65290  FF0A  [OPNRMS] Micro-Cassette: OPEN zum lesen.
65293  FF0D  [MCSMAN] Micro-Cassette: Manueller Betrieb.
65296  FF10  JMP$DFF7 Sprung in den Monitor.
65299  FF13  [REQINI] Sprung    in    die    Initialisierungsroutine
             (Kaltstart).
65302  FF16  [CHKRS]  recover RS232 mode.
65305  FF19  [SNSCOM] Kommando an die Slave-CPU senden.
65308  FF1C  [SRINIT] Serial-Initialisierung.
65311  FF1F  [PWRDWN] Power Fail ("Charge Battery"-Meldung).
65314  FF22  [KYSTST] K(ey)-Speicher aktivieren (initial key stack).
65317  FF25  JMP$DFFD Menue.
65320  FF28  [BINDEC] Umwandlung Binaer in ASCII-Dezimal.
65323  FF2B  [HEXBIN] Umwandlung Hexadezimal in Binaer.
65326  FF2F  [CHKPLG] Pruefung auf angesteckte Optionen.
65329  FF31  [GETCLK] gestrichen.
65332  FF34  [CLOSCS] Cassette: CLOSE.
65335  FF37  [WRITCS] Cassette: Einen Charakter schreiben.
65338  FF3A  [OPNWCS] Cassette: OPEN zum Schreiben.
65341  FF3D  [READCS] Cassette: Einen Charakter lesen.
65344  FF40  [SRCRCS] Cassette: File suchen und OPEN zum lesen.
65347  FF43  [OPNRCS] Cassette: OPEN zum Lesen.
65350  FF46  [PONFCS] Cassette: REMOTE-Buchse ein/aus.
65353  FF49  [DSPLCN] LCD: Bringe n Charakter auf das phys. Display.
65356  FF4C  [DSPLCH] LCD: Einen Charakter auf das phys. Display.
65359  FF4F  JMP$DFF1 LCD: Einen Charakter auf das virt. Display.
65362  FF52  [LCADDR] LCD: Berech. LCD-Chip und Bank aus Koordinaten.
65365  FF55  [LCDMOD] LCD: Kommando zum LCD-Treiber.
65368  FF59  [DATMOD] LCD: Daten oder Adressen zum LCD-Treiber.
```

```
65371   FF5B   [DISPIT] LCD: Charakter ins Display schreiben ohne es  im
               Puffer zu speichern.
65374   FF5E   [DISCON] LCD, TV-Adapter:  Kommandos  und  Daten  zum
               Display-Controller.
65377   FF61   [SCRRCV] LCD: Aktualisiere virt. Display.
65380   FF64   [SOUND]  Ton ausgeben.
65383   FF67   [CHRGEN] Charakter-Generator.
65386   FF6A   [KEYSCN] key scan.
65389   FF6D   [SERIN]  Serial: Input.
65392   FF70   [SEROUT] Serial: Output.
65395   FF73   [SERONF] Serial: Treiber ein/aus.
65398   FF76   [RSPUT]  RS232: Charakter-Sende-Routine.
65401   FF79   [RSGET]  RS232: Charakter-Empfangs-Routine.
65404   FF7C   [RSGSTS] RS232: Empfangs-Status-Routine.
65407   FF7F   [RSCLOS] RS232: Empfangs-CLOSE-Routine.
65410   FF82   [RSOPEN] RS232: Empfangs-OPEN-Routine.
65413   FF85   [RSONOF] RS232: Spannung ein/aus.
65416   FF88   [RSMST]  RS232: Uebertragungsmodus festlegen.
65419   FF8B   [SCRCPY] Drucker: Display-Hardcopy.
65422   FF8E   [PAPFED] Drucker: Papiervorschub.
65425   FF91   [PRTDOT] Drucker: Grafik-Punktreihe.
65428   FF94   [LNPRNT] Drucker: Zeile drucken.
65431   FF97   [CHPRNT] Drucker: Einen Charakter drucken.
65434   FF9A   [KEYIN]  Tastatur: Einen Charakter holen.
65437   FF9D   [KEYSTS] Tastatur: Status.
65440   FFA0   [KYINIT] Tastatur: Initialisierung.
65443   FFA3   [BREKIO] BREAK I/O.
65446   FFA6   [RSTRIO] RESTART I/O.
65449   FFA9   [SLEEP]  SLEEP fuer Menue.
65452   FFAC   [PWROFF] POWER OFF fuer Menue.
65455   FFAF   [CONTIO] i/o continue start.
65458   FFB2   [BREKIN] ON BREAK KEY ENTRY fuer Menue.
65461   FFB5   [CLKINT] clock interrupt initial jump
65464   FFBB   [IRQI8O] Externer IRQ1-Sprung.
65467   FFBB   JMP$DFFA TRAP-Interrupt-Sprung.
65470   FFBE   [SERINT] SCI (RDRF+ORFE+TDRE) Interrupt-Sprung.
65473   FFC1   [TOFINT] TOF (timer overflow) Interrupt-Sprung.
65476   FFC4   [KEYINT] OCF (timer output compare) Interrupt-Sprung.
65479   FFC7   [IRQI8O] ICF (timer input capture) Interrupt-Sprung.
65482   FFCA   [IRQINT] IRQ1 (IS3) Interrupt-Sprung.
65485   FFCD   [INITIO] I/O-Initialisierung.
```

POINTER:

Die angegebene Adresse enthaelt das HIGH BYTE, die folgende Adresse enthaelt das LOW BYTE des Pointers.

```
65488   FFD0   (-> $0145) auf Tastatur-Matrix-Tabelle.
65490   FFD2   (-> $0196) auf Anzahl Daten im 24 Byte-Druckerpuffer.
65492   FFD4   (-> $01E8) auf Anzahl Daten im ext. Cassettenpuffer.
65494   FFD6   (-> $01FF) auf Anzahl Daten im Micro-Cassettenpuffer.
65496   FFD8   (-> $01C2) auf Anzahl Daten im RS232C-Puffer.
65498   FFDA   (-> $0220) auf phys. LCD-Puffer-Adresse.
65500   FFDC   (-> $0378) auf I/O-Puffer-Anfangsadresse (260 Bytes).
65502   FFDE   (-> $027D) auf Speicherplatz fuer Scroll-Geschwindigkeit.
```

65504 FFE0 (-> $02D0) auf ext. Cassettenrecorder Header-Daten.
65506 FFE2 (-> $0324) auf Micro-Cassettenrecorder Header-Daten.
65508 FFE4 (-> $0169) auf Tastaturflag ('keymod').

VERSION-IDENTIFIZIERUNG:

65510 FFE6 Beginn eines Textfeldes (6 ASCII-Zeichen) mit folgender Bedeutung:

'8212KA'	Japan	1.2	10.06.82	made by Kenji Akahane
'8213KA'	Japan	1.3	15.11.82	made by Kenji Akahane
'82U2KA'	USA	1.2	17.06.82	made by Kenji Akahane
'82U3KA'	USA	1.3	15.11.82	made by Kenji Akahane
'82E2KA'	Europa	1.2	10.07.82	made by Kenji Akahane
'82E3KA'	Europa	1.3	15.11.82	made by Kenji Akahane

VEKTOREN-TABELLE:

Die angegebene Adresse enthaelt das HIGH BYTE, die folgende Adresse enthaelt das LOW BYTE des Pointers.

65518 FFEE Zeigt auf 262 ($0106) -> JMP$DFFA (Trap / fuer Monitor)

65520 FFF0 Zeigt auf 265 ($0109) -> JMP$EDD7 (SCI, RDRF+ORFE+TDRE / Interrupt vom seriellen Interface)

65022 FFF2 Zeigt auf 268 ($010C) -> JMP$EF2C (TOF, Timer Overflow / Mikrocassette).

65024 FFF4 Zeigt auf 271 ($010F) -> JMP$F510 (OCF, Timer Output Compare / Keyboard).

65026 FFF6 Zeigt auf 274 ($0112) -> JMP$EF24 (ICF, Timer Input Capture / nicht verwendet).

65028 FFF8 Zeigt auf 277 ($0115) -> JMP$EED6 (IRQ-Vektor / 1.Key Input [K.B.REQUEST Signal] 2.Power On [PWSW Signal] 3.Power Off [PWSW Signal] 4.Clock [CLOCK IRQ Signal] 5.Externer Interrupt [INT EX Signal]).

65030 FFFA Zeigt auf 280 ($0118) SOFTWARE-INTERRUPT (nicht verwendet). Ab hier kann die Adresse fuer den SWI-Vektor selber definiert werden als "JMP xx xx".

65032 FFFC Zeigt auf 283 ($011B) NON MASKABLE INTERRUPT (nicht verwendet). Ab hier kann die Adresse fuer den NMI-Vektor selber definiert werden als "JMP xx xx".

65034 FFFE Zeigt auf 57344 ($E000). RESET-Vektor.

BESCHREIBUNG DER ROM-ROUTINEN

Die Register werden wie folgt bezeichnet:

(A) Akkumulator A
(B) Akkumulator B
(X) Indexregister X
(C) Carry Flag
(Z) Zero Flag

RS 232 C

UEBERTRAGUNGSMODUS FESTLEGEN [RSMST] $FF88

```
Einsprung: (A) Ø,1:  Stop Bits (1 oder 2)
               2:    CARRIER DETECT (Ø:check 1:ignore)
               3:    RTS  (Ø:low 1:high)
               4:    DSR  (Ø:check 1:ignored)
               5:    CTS  (Ø:check 1:ignored)
               7,6:  Parity (ØØ:even 1Ø:odd 11:none)

           (B) LSB4: Bit-Laenge (5,6,7 oder 8)
               MSB4: Baud-Rate
                     Ø: 11Ø BPS
                     1: 15Ø BPS
                     2: 3ØØ BPS
                     3: 6ØØ BPS
                     4:12ØØ BPS
                     5:24ØØ BPS
                     6:48ØØ BPS
                     7:96ØØ BPS (nicht verfuegbar)

Register:  A,B,X
```

SPANNUNG EIN/AUS [RSONOF] $FF85

```
Einsprung: (A) Ø:aus  <>Ø:ein

Rueckkehr: (C) Ø:normal ein  1:Fehler
           (A) Ø:normal  <>Ø:Fehlercode
           (Z) entsprechend Wert von (A)

Register:  B,X
```

OPEN [RSOPEN] $FF82

```
Einsprung: (X) Puffer-Adresse
           (A,B) Puffer-Laenge (1...)

Rueckkehr: (C) Ø:normal  1:I/O-Fehler
           (A) Ø:o.k.  <>Ø:Fehler-Code
           (Z) entsprechend Wert von (A)

Register:  B,X
```

CLOSE [RSCLOS] $FF7F

Einsprung: keine Parameter

Rueckkehr: (C) Ø:normal 1:I/O-Fehler
(A) Ø:o.k. <>Ø:Fehler-Code
(Z) entsprechend Wert von (A)

Register: B,X

EMPFANGEN EINES CHARAKTERS [RSGET] $FF79

Empfaengt einen Charakter vom RS232C-Puffer. Diese Routine wartet nicht auf Sendedaten, auch wenn der Puffer leer ist.

Einsprung: keine Parameter

Rueckkehr: (A) Daten
(B) Status $ØØ:empfangen
$Ø1:Puffer leer
MSBl=1:Fehler
BitØ-6:Fehler-Code
(C) Ø:normal 1:I/O-Fehler
(Z) entsprechend Wert von (B)

Register: X

SENDEN EINES CHARAKTERS [RSPUT] $FF76

Einsprung: (A) Daten

Rueckkehr: (B) Status Bit Ø (1:DSR aus)
Bit 1 (1:CTS aus)
Bit 2-7 immer Ø
(Z) entsprechend Wert von (B)

Register: X,A

Wenn (B) nicht Ø ist, wurden die Daten nicht uebertragen!

RS232C-PROZEDUR

1. Uebertragungsmodus festlegen (CALL 'RSMST')
2. Treiber einschalten (CALL 'RSONOF')
3. OPEN (fuer lesen)
4. senden/empfangen
5. CLOSE

Wenn man nach dem Oeffnen zum Lesen, den Uebertragungsmodus neu definiert, aendert sich nur der Modus zum Senden, der Empfangsmodus bleibt erhalten.

Wenn ein Peripheriegeraet der Slave-CPU betrieben wird (Drucker, Cassette, Tongeber) gehen uebertragene RS232C-Daten verloren.

Nach 'RSOPEN' werden eingehende Daten in den RS232C-Puffer geschrieben.

TONGEBER

TON AUSGEBEN [SOUND] $FF64

Einsprung: (A) Ton
Ø:Pause
1:DO 2:RE 3:MI 4:FA 5:SO 6:RA (44Ø Hz) .. 28:TI
19:#2 3Ø:#3 31:#4 .. 56:#28
(B) Laenge
Ø:kein Ton
1:Ø.1 Sek. .. 255:25.5 Sek.

Rueckkehr: (C) Ø:normal 1:Fehler

Register: A,B,X

DRUCKER

EINEN CHARAKTER DRUCKEN [CHPRNT] $FF97

Einsprung: (A) Charakter-Code (ASCII)

Rueckkehr: (C) Ø:normal 1:I/O-Fehler

Register: A,B,X

Charakter werden erst ausgedruckt, wenn 'Line Feed ' ($ØA) empfangen wird, oder 24 Charakter im Druckerpuffer sind. Contol-Code 'Carriage Return' ($ØD) setzt den Puffer auf den Anfang.

EINE ZEILE DRUCKEN [LNPRNT] $FF94

Einsprung: (X) Puffer-Adresse (24 Byte Laenge)

Rueckkehr: (C) Ø:normal 1:I/O-Fehler

Register: A,B,X

EINE GRAFIK-PUNKTREIHE DRUCKEN [PRTDOT] $FF91

Einsprung: (X) Puffer-Adresse (24 Byte Laenge)

Rueckkehr: (C) Ø:normal 1:I/O-Fehler

Register: A,B,X

Puffer- Byte Ø: 1. 6 Punkte ...
format: Byte 23: 24. 6 Punkte

```
    jeweils Bit   Ø:  1. Punkt ...
            Bit   5:  6. Punkt
            Bit 6,7: nicht verwendet
```

DISPLAY-HARDCOPY [SCRCPY] $FF8B

```
Einsprung: keine Parameter

Rueckkehr: (C) Ø:normal  1:I/O-Fehler

Register:  A,B,X
```

PAPIERVORSCHUB [PAPFED] $FF8E

```
Einsprung: (A) Anzahl Punktreihen (1...255)

Rueckkehr: (C) Ø:normal  1:I/O-Fehler

Register:  A,B,X
```

EXTERNER CASSETTENRECORDER

REMOTE-BUCHSE EIN/AUS [PONFCS] $FF46

```
Einsprung: (A) Ø:aus  1:ein

Rueckkehr: (C) Ø:normal  1:Fehler
```

OPEN ZUM LESEN [OPNRCS] $FF43

```
Sucht Zielfile ohne es anzuzeigen.

Einsprung: (X) Paket-Adresse

        Paket  Byte Ø:   Lese-Modus
                         ØØ:bei jedem Block stoppen
                         Ø1:nicht stoppen
                         FF:abhaengig vom Header
               Byte 1:   Puffer-Adresse (high)
               Byte 2:   Puffer-Adresse (low)
                         Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ

Rueckkehr: (A) Ø:normal  <>Ø:Fehler
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)
```

OPEN ZUM LESEN MIT FILE-NAMEN-ANGABE [SRCRCS] $FF4Ø

```
Zeigt gefundene Filenamen auf dem Display an.
```

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø:   Lese-Modus
                         ØØ:bei jedem Block stoppen
                         Ø1:nicht stoppen
                         FF:abhaengig vom Header
               Byte 1:   Puffer-Adresse (high)
               Byte 2:   Puffer-Adresse (low)
                         Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ
               Byte19-26:gefundener File-Name (bei Rueckkehr)
               Byte27-34:gefundener File-Typ  (bei Rueckkehr)

Rueckkehr: (A) ØØ:normal  $8Ø:falsches File ("skipped")
               anderer Code:Fehler-Code
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)
```

EINEN CHARAKTER LESEN [READCS] $FF3D

```
Einsprung: keine Parameter

Rueckkehr: (A) Charakter-Code
           (B) Fehler-Code (Ø:kein Fehler)
           (Z) entsprechend Wert von (B)
           (C) Ø:normal  1:Fehler

Register:  X
```

CLOSE [CLOSCS] $FF34

```
Einsprung: keine Parameter

Rueckkehr: (C) Ø:normal  1:Fehler
```

OPEN ZUM SCHREIBEN [OPNWCS] $FF3A

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø    :Schreib-Modus
                          ØØ jeden Block stoppen
                          Ø1 nicht stoppen
               Byte 1:    Puffer-Adresse (high)
               Byte 2:    Puffer-Adresse (low)
                          Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ

Rueckkehr: (A) Ø:normal  <>Ø:Fehler
           (Z) entsprechend Wert von (A)
           (C) Ø:normal  1:Fehler
```

EINEN CHARAKTER SCHREIBEN [WRITCS] $FF37

```
Einsprung: (A) Charakter

Rueckkehr: (B) Ø:normal  <>Ø:Fehler
           (Z) entsprechend Wert von (B)
           (C) Ø:normal  1:Fehler

Register:  X
```

EINGEBAUTER MICRO-CASSETTENRECORDER

OPEN ZUM LESEN [OPNRMS] $FFØA

Sucht Zielfile ohne Files anzuzeigen.

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø:   Lese-Modus
                         ØØ:bei jedem Block stoppen
                         Ø1:nicht stoppen
                         FF:abhaengig vom Header
               Byte 1:   Puffer-Adresse (high)
               Byte 2:   Puffer-Adresse (low)
                         Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ

Rueckkehr: (A) Ø:normal  <>Ø:Fehler
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)
```

OPEN ZUM LESEN MIT FILE-NAMEN-ANGABE [SRCRMS] $FFØ7

Zeigt gefundene Filenamen auf dem Display an.

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø:   Lese-Modus
                         ØØ:bei jedem Block stoppen
                         Ø1:nicht stoppen
                         FF:abhaengig vom Header
               Byte 1:   Puffer-Adresse (high)
               Byte 2:   Puffer-Adresse (low)
                         Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ
               Byte19-26:gefundener File-Name (bei Rueckkehr)
               Byte27-34:gefundener File-Typ  (bei Rueckkehr)

Rueckkehr: (A) ØØ:normal  $8Ø:falsches File ("skipped")
               anderer Code:Fehler-Code
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)
```

EINEN CHARAKTER LESEN [READMS] $FFØ4

```
Einsprung: keine Parameter

Rueckkehr: (A) Charakter-Code
           (B) Fehler-Code (Ø:kein Fehler)
           (Z) entsprechend Wert von (B)
           (C) Ø:normal  1:Fehler

Register:  X
```

CLOSE [CLSMCS] $FEFB

```
Einsprung: keine Parameter

Rueckkehr: (C) Ø:normal  1:Fehler
```

OPEN ZUM SCHREIBEN [OPNWMS] $FFØ1

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø   :Schreib-Modus
                         ØØ jeden Block stoppen
                         Ø1 nicht stoppen
               Byte 1:   Puffer-Adresse (high)
               Byte 2:   Puffer-Adresse (low)
                         Puffer-Laenge = 256+4 Bytes
               Byte 3-1Ø:File-Name
               Byte11-18:File-Typ

Rueckkehr: (A) Ø:normal  <>Ø:Fehler
           (Z) entsprechend Wert von (A)
           (C) Ø:normal  1:Fehler
```

EINEN CHARAKTER SCHREIBEN [WRITMS] $FEFE

```
Einsprung: (A) Charakter

Rueckkehr: (B) Ø:normal  <>Ø:Fehler
           (Z) entsprechend Wert von (B)
           (C) Ø:normal  1:Fehler

Register:  X
```

AN BANDANFANG SPULEN [REWMCS] $FEF5

```
Einsprung: kein Parameter

Rueckkehr: (C) Ø:normal  1:Fehler
           (A) Ø:normal  <>Ø:Fehler-Code
           (Z) entsprechend Wert von (A)
```

NACH ZAEHLWERKSTAND SPULEN [SEKMCS] $FEF2

```
Einsprung: (X) gewuenschter Zaehlwerkstand

Rueckkehr: (A) Ø:normal  1:Cassettenfehler
               2:Bandanfang  3:Bandende

Register:  keine
```

ZAEHLWERK LESEN/SCHREIBEN [CNTMCS] $FEEF

```
Einsprung: (A) Ø:Lesen  1:Setzen
           (X) Zaehlwerkstand (beim Setzen)

Rueckkehr: (X) Zaehlwerkstand (beim Lesen)

Register:  B
```

INT./EXT. CASSETTEN-LESE/SCHREIB-PROZEDUR

```
SCHREIBEN: 1. OPEN zum Schreiben
           2. Schreiben
           3. CLOSE

LESEN:     1. OPEN zum Lesen (eine der beiden Routinen)
           2. Lesen
           3. CLOSE
```

PROM-CARDRIDGE

OPEN ZUM LESEN [OPNPRM] $FEF2

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø- 7:File-Name
               Byte 8-15:File-Typ
               Byte16-23:gefundener File-Name (nach READ)
               Byte24-31:gefundener File-Typ  (nach READ)

           (A) Ø:READ-MODE keine Anzeige der File-Namen
               1:Anzeige der File-Namen

Rueckkehr: (A) $ØØ:normal  $AØ:keine PROM-Cassette
               $A1:nicht gefunden  $A2:OPEN-Fehler
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)
```

EINEN CHARAKTER LESEN [REDPRM] $FEE9

```
Einsprung: keine Parameter

Rueckkehr: (A) Charakter-Code
           (B) $ØØ:normal  $Ø1:End of File  andere:Fehler
           (C) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (B)
```

```
Register:  X
```

CLOSE [CLSPRM] $FEE6

```
Einsprung: keine Parameter

Rueckkehr: (C) Ø:normal  1:Fehler

Register:  keine
```

DIRECTORY LESEN [DIRPRM] $FEE3

```
Einsprung: (X) Adresse, wohin Dir.-Information gespeichert wird
           (A) Directory-Nummer (Ø...63)

Rueckkehr: (C) Ø:normal  1:Fehler
           (A) Ø:normal  1:Fehler
           (Z) entsprechend Wert von (A)

Register:  keine

Directory- Byte Ø- 7:File-Name
Format:    Byte 8-15:File-Typ
           Byte16-19:Start-Adresse
           Byte2Ø-23:End-Adresse +1
           Byte24-29:Datum
           Byte3Ø-31:nicht zugeordnet
```

PRUEFUNG AUF ANGESTECKTE OPTIONEN [CHKPLG] $FF2F

```
Setzt den Wert in Adresse $ØØ79.

Einsprung: keine Parameter

Rueckkehr: (A) Bit 2,1,Ø: ØØØ PROM-Cassette
                          ØØ1 Reserve
                          Ø1Ø Nichts angesteckt
                          Ø11 Reserve
                          1xx Micro-Cassette
               Bit 3 - 7: Ø
           (Z) entsprechend Wert von (A)

Register:  B,X
```

MONITOR READ/WRITE (binary dump)

OPEN ZUM DUMPEN DES MEMORY-INHALTS [OPNDMP] $FEEØ

```
Einsprung: (X) Paket-Adresse

        Paket  Byte Ø   : Schreib-Modus
                          ØØ jeden Block stoppen
                          Ø1 nicht stoppen
```

```
                Byte 1    : Puffer-Adresse (high)
                Byte 2    : Puffer-Adresse (low)
                Byte 3-1Ø: File-Name
                Byte11-18: File-Typ
                Byte19-2Ø: Dump-Startadresse
                Byte21-22: Dump-Endadresse
                Byte23-24: Dump-Ofset
                Byte25-26: RUN-Adresse
            (B) Geraetenummer

Rueckkehr:  (A) Ø:normal   <>Ø:Fehler
            (Z) entsprechend Wert von (A)
            (C) Ø:normal   1:I/O-Fehler
```

DUMP MEMORY-INHALT UND AUTOMATISCHES CLOSE [BIDUMP] $FEDD

```
Einsprung:  (B) Geraetenummer

Rueckkehr:  (A) Ø:normal   <>Ø:Fehler
            (Z) entsprechend Wert von (A)
            (C) Ø:normal   1:I/O-Fehler
```

OPEN ZUM LESEN (READ) DES MEMORY-INHALTS [OPNLOD] $FEDA

```
Einsprung:  (A) 1:File-Name bei Rueckkehr
                Ø:kein File-Name
            (X) Paket-Adresse

          Paket Byte Ø    : $ØØ
                Byte 1    : Puffer-Adresse (high)
                Byte 2    : Puffer-Adresse (low)
                            Puffergroesse = 256 + 4 Bytes
                Byte 3-1Ø: File-Name
                Byte11-18: File-Typ

            (B) Geraetenummer

Rueckkehr:  (A) Ø:normal   <>Ø:Fehler
            (Z) entsprechend Wert von (A)
            (C) Ø:normal   1:I/O-Fehler
```

LESEN DES MEMORY INHALTS (READ) UND AUTOM. CLOSE [BILOAD] $FED7

```
Einsprung:  (A) ØØ: laden mit CRC verify
                Ø1: verify, nicht laden
            (B) Geraete-Nummer
            (X) Offset-Wert

Rueckkehr:  (A) Ø:normal   <>Ø:Fehler
            (Z) entsprechend Wert von (A)
            (C) Ø:normal   1:I/O-Fehler
            (X) Startadresse (Offset addieren)
```

PROZEDUR

```
WRITE:      1. OPEN zum Schreiben (dump)
            2. Schreiben
```

```
READ:       1. OPEN zum Lesen (read)
            2. Lesen

Geraetenr.:'Ø' RS232C  11Ø BPS  ('Ø'...'6' nur in
           '1' RS232C  15Ø BPS   der USA-Version
           '2' RS232C  3ØØ BPS   verfuegbar.)
           '3' RS232C  6ØØ BPS
           '4' RS232C 12ØØ BPS
           '5' RS232C 24ØØ BPS
           '6' RS232C 48ØØ BPS
           'M' Micro-Cassette
           'C' externe Cassette
           'P' PROM-Cardridge
```

SERIAL

TREIBER EIN/AUS [SERONF] $FF73

```
Einsprung: (A) Ø:aus  1:ein
Rueckkehr: (C) Ø:normal  1:I/O-Fehler
           (A) Ø:normal  <>Ø:Fehler
           (Z) entsprechend Wert von (A)

Register:  B,X
```

SERIAL OUTPUT [SEROUT] $FF7Ø

```
Einsprung: (X) Paket-Adresse

        Paket  Byte 1:Format
               Byte 2:Empfangs-Geraetenummer
               Byte 3:Sende-Geraetenummer
               Byte 4:Funktion
               Byte 5:Charakterlaenge (Datenstring -1)
               .
               .
               Byte N:Letzter Charakter

           (A) LSB=1:Nach Senden Empfangsfunktion einschalten
               LSB=Ø:Empfangsfunktion nicht einschalten

Rueckkehr: (C) Ø:normal  1:I/O-Fehler
           (A) Ø:normal  $B1:Geraete-Fehler

Register:  X
```

```
BEISPIEL:  LDAA #$ØØ
           LDX  #PAKET
           JSR  SEROUT
           .
           .
           SEROUT EQU $FF7Ø
           PAKET  FCB $Ø,$31,$3Ø,$34,$ØD  * sendet 'CR' zum Monitor
```

EMPFANGEN VOM SERIAL-PORT [SERIN] $FF6D

```
Einsprung: (X) Adresse des empfangenen Daten-Strings

Rueckkehr: (C) Ø:normal  1:I/O-Fehler
           (A) Ø:normal  $BØ:time out  $B1:Geraete-Fehler
           (B) Empfangener Block-Status
               Ø:empfangen mit Header
               1:empfangen ohne Header

Register:  X


EMPFANGENE DATEN (MIT HEADER):
           1. Format
           2. Empfangs-Geraete-Nummer (1 Byte)
           3. Sende-Geraete-Nummer (1 Byte)
           4. Funktion (1 Byte, gesetzt nach Uebertragung)
           5. Paket-Charakter-Laenge (1 Byte, Datenstring - 1,
              gesetzt nach Uebertragung)
           .  gesetzt nach Uebertragung
           .
           N. letzter Charakter

EMPFANGENE DATEN (OHNE HEADER):
           1. nicht benutzt
           2. nicht benutzt
           3. nicht benutzt
           4. nicht benutzt
           5. Paket-Charakter-Laenge (1 Byte, Datenstring - 1,
              gesetzt nach Uebertragung)
           .  gesetzt nach Uebertragung
           .
           N. letzter Charakter


BEISPIEL:  LDX #RCVADR
           JSR SERIN
           .
           .
           SERIN  EQU $FF6D
           RCVADR RMB 1  * Format
                  RMB 1  * Destin. Dev. Number
                  RMB 1  * Source  Dev. Number
                  RMB 1  * Function
                  RMB 5  * Received Data
```

LCD-ROUTINEN

BERECHNEN VON LCD-CHIP UND BANK AUS KOORDINATEN [LCADDR] $FF52

```
Einsprung: (A) Punkt Position (Ø...119d)
           (B) Zeile (Ø...31d)

Rueckkehr: (A) Adresse im LCD-Treiber
           (B) LCD-Chip-Nummer (Bit3=1)
```

Register: X

DATEN ZUM LCD-TREIBER [LCDMOD] $FF5D

Einsprung: (A) Daten (Kommando oder Daten)

Register: A,B,X

NACH LCD-READY-CHECK RUFE 'WRTP26' [DATMOD] $FF59

Einsprung: (A) Ziel-Bit-Position
(B) Daten (entsprechen 'WRTP26')

Register: A,B,X

CHARAKTER INS DISPLAY SCHREIBEN, NICHT IN PUFFER [DISPIT] $FF5B

Der Charakter wird nicht in den Display-Pufferspeicher geschrieben, verschwindet also evtl. beim Scrollen wieder.

Einsprung: (A) Charakter-Code (ASCII)
(X) LCD-Adresse (high:Spalte, low:Zeile)

Rueckkehr: (X) naechste Adresse

Register: keine

EINEN CHARAKTER INS DISPLAY BRINGEN, AUCH IN PUFFER [DSPLCH] $FF4C

Einsprung: (A) Charakter-Code (ASCII)
(X) LCD-Adresse (high:Spalte, low:Zeile)

Rueckkehr: (X) naechste Adresse

Register: keine

EINEN CHARAKTER INS VIRT.DISPLAY BRINGEN $FF4F

Einsprung: (A) Charakter-Code (ASCII)

N CHARAKTER INS DISPLAY BRINGEN, AUCH IN PUFFER [DSPLCN] $FF49

Einsprung: (B) Charakter-Anzahl (Ø=Display loeschen)
(X) Adresse des Daten-Strings
Register: X,B (wenn B<> Ø)

KOMMANDOS UND DATEN ZUM DISPLAY CONTROLER [DISCON] $FF5E

Einsprung: (X) Paket-Adresse

Paket Ø: Funkton
1: Daten 1

```
                .
                .
               N: Daten N

Rueckkehr:  Paket enthaelt Daten der naechsten Funktions-Adresse.

Register:  X,B


DATENPAKETE FUER VERSCHIEDENE KOMMANDOS:

FMT = Format-Code (ØØ:Aufruf Ø1:Rueckkehr)
DID = Empfangender Prozessor
SID = Sendender Prozessor
FNC = Funktionscode
SIZ = Datenfeldlaenge -1

MM = Master ID
SS = Slave ID
XX = 'don't care'-Byte
NN = spezifizierter Wert

Alle numerischen Werte hexadezimal.


FMT DID SID FNC SIZ  Funktion (L:LCD  C:CRT=Monitor)
-------------------------------------------------------------------

                     GERAETE-AUSWAHL (L,C)
ØØ   MM   SS   84   ØØ   ØØ-ØØ = Geraet (LCD:22 Monitor:3Ø)
Ø1   SS   MM   84   ØØ   ØØ-ØØ = Fehler-Code
                              ØØ: kein Fehler
                              FE: Geraet nicht bereit
                              FF: Geraetename falsch

                     GERAET INITIALISIEREN (C)
ØØ   MM   SS   85   ØØ   ØØ-ØØ = XX
Ø1   SS   MM   85   ØØ   ØØ-ØØ = Fehler-Code
                              ØØ: kein Fehler
                              FF: I/O-Fehler

                     GERAET CHECKEN UND PARAMETER HOLEN (L,C)
ØØ   MM   SS   86   ØØ   ØØ-ØØ = XX
Ø1   SS   MM   86   ØØ   ØØ-ØØ = Geraete-Nummer
                     Ø1-Ø2 = Schirm-Anfangsadresse (C)
                     Ø3-Ø4 = max. Schirmgroesse (C)

                     DISPLAY-PARAMETER DEFINIEREN (L,C)
ØØ   MM   SS   87   Ø3   ØØ-ØØ = Zeilenbreite des virt. Schirms
                     Ø1-Ø1 = Anzahl Zeilen des virt. Schirms
                     Ø2-Ø3 = Anfangsadresse virt. Schirm (nicht verw.)
Ø1   SS   MM   87   ØØ   ØØ-ØØ = Fehler-Code
                              ØØ: kein Fehler
                              FF: Parameter nicht korrekt
                              FE: Anfangsadresse nicht korrekt

                     DISPLAY-PARAMETER LESEN  (L,C)
ØØ   MM   SS   88   ØØ   ØØ-ØØ = XX
```

```
Ø1   SS   MM   88   Ø1    ØØ-ØØ = Zeilenbreite
                          Ø1-Ø1 = Anzahl Zeilen

                          PHYSIKALISCHE DISPLAY-GROESSE LESEN  (L,C)
ØØ   MM   SS   89   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   89   Ø1    ØØ-ØØ = Zeilenbreite
                          Ø1-Ø1 = Anzahl Zeilen

                          PHYS.SCREEN-POINTER AUF VIRT.SCHIRM SETZEN (L,C)
ØØ   MM   SS   CØ   Ø1    ØØ-ØØ = X-Koordinate
                          Ø1-Ø1 = Y-Koordinate
Ø1   SS   MM   CØ   ØØ    ØØ-ØØ = XX

                          PHYS.SCREEN-POINTER AUF VIRT.SCHIRM LESEN (L,C)
ØØ   MM   SS   8A   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   8A   Ø1    ØØ-ØØ = X-Koordinate
                          Ø1-Ø1 = Y-Koordinate

                          CURSORPOSITION AUF VIRT.SCHIRM SETZEN (L,C)
ØØ   MM   SS   C2   Ø1    ØØ-ØØ = X-Koordinate
                          Ø1-Ø1 = Y-Koordinate
Ø1   SS   MM   C2   ØØ    ØØ-ØØ = XX

                          CURSORPOSITION AUF VIRT.SCHIRM LESEN (L,C)
ØØ   MM   SS   8C   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   8C   ØØ    ØØ-ØØ = X-Koordinate
                          Ø1-Ø1 = Y-Koordinate

                          CURSOR-GRENZE SETZEN (L,C)
ØØ   MM   SS   C3   ØØ    ØØ-ØØ = Grenze
Ø1   SS   MM   C3   ØØ    ØØ-ØØ = XX

                          CURSOR-GRENZE LESEN (L,C)
ØØ   MM   SS   C3   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   8D   ØØ    ØØ-ØØ = Grenze

                          SCROLL-WEITE SETZEN (L,C)
ØØ   MM   SS   C4   Ø1    ØØ-ØØ = horizotale Weite
                          Ø1-Ø1 = vertikale Weite
Ø1   SS   MM   C4   ØØ    ØØ-ØØ = XX

                          SCROLL-WEITE LESEN (L,C)
ØØ   MM   SS   8E   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   8E   Ø1    ØØ-ØØ = horizotale Weite
                          Ø1-Ø1 = vertikale Weite

                          LIST-FLAG SETZEN (L,C)
ØØ   MM   SS   C5   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   C5   ØØ    ØØ-ØØ = XX

                          LIST-FLAG LOESCHEN (L,C)
ØØ   MM   SS   C6   ØØ    ØØ-ØØ = XX
Ø1   SS   MM   C6   ØØ    ØØ-ØØ = XX

                          PUNKT SETZEN (L,C)
ØØ   MM   SS   C7   Ø4    ØØ-Ø1 = X-Koordinate
                          Ø2-Ø3 = Y-Koordinate
                          Ø4-Ø4 = Farbcode (Ø=PRESET)
Ø1   SS   MM   C7   ØØ    ØØ-ØØ = XX
```

```
                                PUNKT ABFRAGEN (L,C)
ØØ   MM   SS   8F   Ø3          ØØ-Ø1 = X-Koordinate
                                Ø2-Ø3 = Y-Koordinate
Ø1   SS   MM   8F   ØØ          ØØ-ØØ = Farbcode (Ø=nicht gesetzt)

                                LINIE ZEICHNEN (L,C)
ØØ   MM   SS   C8   Ø8          ØØ-Ø1 = X-Koordinate Startpunkt
                                Ø2-Ø3 = Y-Koordinate Startpunkt
                                Ø4-Ø5 = X-Koordinate Endpunkt
                                Ø6-Ø7 = Y-Koordinate Endpunkt
                                Ø8-Ø8 = Farbcode (Ø=PRESET)
Ø1   SS   MM   C8   ØØ          ØØ-ØØ = XX

                                BREITE DER AKTUELLEN ZEILE LESEN (L,C)
ØØ   MM   SS   91   ØØ          ØØ-ØØ = XX
Ø1   SS   MM   91   Ø3          ØØ-ØØ = erste Zeilennummer der aktuellen Zeile
                                Ø1-Ø1 = letzte Zeilennummer der aktuellen Zeile
                                Ø2-Ø2 = letzte X-Koordinate
                                Ø3-Ø3 = letzte Y-Koordinate

                                ZEILENABSCHLUSS SETZEN (L,C)
ØØ   MM   SS   C9   ØØ          ØØ-ØØ = Zeilennummer
Ø1   SS   MM   C9   ØØ          ØØ-ØØ = XX

                                EINEN CHARAKTER IN VIRT.SCREEN SCHREIBEN (L,C)
ØØ   MM   SS   92   ØØ          ØØ-ØØ = Charakter-Code
Ø1   SS   MM   92   Ø1          ØØ-ØØ = X-Cursorposition
                                Ø1-Ø1 = Y-Cursorposition

                                GRAPHIC-SCREEN LOESCHEN (L,C)
ØØ   MM   SS   CA   ØØ          ØØ-ØØ = Hintergrundfarbe (C)
Ø1   SS   MM   CA   ØØ          ØØ-ØØ = XX

                                SCROLL-GESCHWINDIGKEIT SETZEN (C)
ØØ   MM   SS   CB   ØØ          ØØ-ØØ = Scroll-Geschwindigkeit (Ø...9)
Ø1   SS   MM   CB   ØØ          ØØ-ØØ = XX

                                DISPLAY-MODUS SETZEN (C)
ØØ   MM   SS   93   Ø2          ØØ-ØØ = Charakter-Modus
                                        ØØ: inaktiv
                                        Ø1: aktiv
                                Ø1-Ø1 = Grafik-Modus
                                        ØØ: inaktiv
                                        Ø1: Modus Ø

                                        Ø2: Modus 1

                                Ø2-Ø2 = Hintergrundfarbe
                                        ØØ: Gruen
                                        Ø1: Gelb
                                        Ø2: Blau
                                        Ø3: Rot
                                        Ø4: Weiss
                                        Ø5: Cyan
                                        Ø6: Magenta
                                        Ø7: Orange
Ø1   SS   MM   93   ØØ          ØØ-ØØ = Fehler-Code
                                        ØØ: kein Fehler
                                        FF: Fehler
```

```
                                ?CURSORPOSITION AUF PHYS.SCHIRM SETZEN (C)
ØØ   MM   SS   CC   Ø1          ØØ-ØØ = X-Koordinate
                                Ø1-Ø1 = Y-Koordinate
Ø1   SS   MM   CC   ØØ          ØØ-ØØ = XX

                                ?CURSORPOSITION AUF PHYS.SCHIRM LESEN (C)
ØØ   MM   SS   94   ØØ          ØØ-ØØ = XX
Ø1   SS   MM   94   Ø1          ØØ-ØØ = X-Koordinate
                                Ø1-Ø1 = Y-Koordinate

                                EINEN CHARAKTER AN ACC-POSITION AUF PHYS.SCREEN
ØØ   MM   SS   CD   Ø1          ØØ-ØØ = Charakter-Code
                                Ø1-Ø1 = Farb-Code
Ø1   SS   MM   CD   ØØ          ØØ-ØØ = XX

                                ACC-POSITION AUF PHYS.SCREEN SETZEN (C)
ØØ   MM   SS   CD   Ø1          ØØ-ØØ = X-Koordinate
                                Ø1-Ø1 = Y-Koordinate
Ø1   SS   MM   CE   ØØ          ØØ-ØØ = XX

                                EINEN CHARAKTER AUF ACC-POSITION LESEN (C)
ØØ   MM   SS   95   ØØ          ØØ-ØØ = XX
Ø1   SS   MM   95   Ø1          ØØ-ØØ = Charakter-Code
                                Ø1-Ø1 = Farbcode

                                FARBSATZ-AUSWAHL (C)
ØØ   MM   SS   CF   ØØ          ØØ-ØØ = Farbsatz
                                        ØØ: Satz Ø
                                        Ø1: Satz 1
Ø1   SS   MM   CF   ØØ          ØØ-ØØ = XX

                                CURSOR-MODUS SETZEN (C)
ØØ   MM   SS   DØ   ØØ          ØØ-ØØ = Cursor-Modus
                                        ØØ: ein
                                        Ø1: aus
Ø1   SS   MM   DØ   ØØ          ØØ-ØØ = XX

                                EINE ZEILE (32 CHAR.) LESEN AB READ-POINTER (C)
ØØ   MM   SS   96   ØØ          ØØ-ØØ = XX
Ø1   SS   MM   96   2Ø          Ø1-2Ø = Charakter-Codes

                                GRAPHIK-CHARAKTER DEFINIEREN (C)
ØØ   MM   SS   D1   Ø6          ØØ-ØØ = Charakter-Code
                                Ø1-Ø6 = Charakter-Generator-Muster
Ø1   SS   MM   D1   Ø1          ØØ-ØØ = XX

                                ?KREIS ZEICHNEN (C)
ØØ   MM   SS   D2   Ø?          ØØ-Ø? =
Ø1   SS   MM   D2   ØØ          ØØ-ØØ = XX

                                ?MALEN (C)
ØØ   MM   SS   D3   Ø?          ØØ-Ø? =
Ø1   SS   MM   D3   ØØ          ØØ-ØØ = XX


*** NEUE KOMMANDOS (1982/Ø5/Ø8) ***
```

```
                          NN CHARAKTER VOM VIRT.SCREEN LESEN  (L,C)
ØØ  MM  SS  97  Ø3        ØØ-ØØ = Anfangs-X-Koordinate
                          Ø1-Ø1 = Anfangs-Y-Koordinate
                          Ø2-Ø3 = Anzahl Charakter
Ø1  SS  MM  97  NN        ØØ-NN = Gelesene Charakter-Codes

                          EINEN CHARAKTER AUF VIRT.SCREEN SCHREIBEN (L,C)
ØØ  MM  SS  98  ØØ        ØØ-ØØ = Charakter-Code
Ø1  SS  MM  98  Ø3        ØØ-ØØ = Neue X-Cursorposition
                          Ø1-Ø1 = Neue Y-Cursorposition
                          Ø2-Ø2 = Erste Zeilennummer
                          Ø3-Ø3 = Letzte Zeilennummer
```

BEISPIEL

Das Indexregister enthaelt die Anfangsadresse eines Datenpaketes, dessen Bedeutung je nach Kommando unterschiedlich ist. Die Formen sind oben erlaeutert, Byte Ø enthaelt immer den Funktionscode (Spalte FNC), die folgenden Bytes enthalten die Daten.

Zum Beispiel hat der Befehl 'PUNKT SETZEN' den Funktionscode 'C7', als Daten dienen die X- und Y-Koordinaten, sowie der Farbcode (Ø entspricht PRESET=loeschen).

Wenn man den Punkt auf die Koordinaten X=$ØØ4Ø und Y=$ØØ1Ø setzen moechte und die Paket-Anfangsadresse (X-Register) auf $ØA4Ø gesetzt hat, sieht das Paket folgendermassen aus:

```
$ØA4Ø: C7
$ØA41: ØØ 4Ø
$ØA43: ØØ 1Ø
$ØA45: Ø1
```

Die anderen Funktionen arbeiten analog, nur mit anderen Parametern, nach der Abarbeitung der Routine befinden sich etwaige Daten nach dem Funktionscode im Datenpaket.

AKTUALISIERE VIRTUELLES DISPLAY [SCRRCV] $FF61

Einsprung: keine Parameter

Register: keine

FEHLER-CODES BEI RUECKKEHR

EXTERNE- UND MICRO-CASSETTE

```
$Ø1:  Ende des Files
$8Ø:  Keine Micro-Cassette vorhanden
$81:  Lesefehler
$82:  File nicht gefunden
$83:  CRC Fehler
```

```
$84:  READ ohne OPEN
$85:  File-Fehler
$86:  READ(binaer)-Fehler
$87:  CLOSE ohne OPEN
$88:  Doppeltes OPEN
$89:  Lese-Puffer voll
$8A:  READ-CHECK-Fehler
$8B:  File gefunden, aber nicht gewuenschtes
$8C:  binary load section error
$91:  WRITE-Fehler
$94:  WRITE ohne OPEN
$99:  READ/WRITE: Falsche Geraete-Nummer
```

PROM-CARDRIDGE

```
$AØ:  Keine PROM-Cassette vorhanden
$A1:  File nicht gefunden
$A2:  OPEN-Fehler
$A3:  Falsche Directory-Nummer
$A4:  PROM-Header-Format falsch
$A5:  Falsche Adresse
```

SERIAL COMMUNICATION

```
$BØ:  Zeit ueberschritten
$B1:  Geraete-Fehler
$B2:  Kommunikations-Fehler
$B3:  Treiber nicht eingeschaltet
```

RS 232 C

```
$CØ:  Paritaets-Fehler
$C1:  CD-Fehler
```

TASTATUR

TASTATUR-STATUS LESEN [KEYSTS] $FF9D

```
Einsprung: keine Parameter

Rueckkehr: (A) Anzahl Charakter im Tastaturpuffer
           (C) Ø:normal  1:Fehler

Register:  B,X
```

CODE AUS TASTATURPUFFER LESEN [KEYIN] $FF9A

```
Einsprung: keine Parameter

Rueckkehr: (A) Charakter-Code
               $FE = Funktionstaste
           (B) Funktionscode wenn $FE in (A)
           (C) Ø:normal  1:Fehler
```

Register: X,B (wenn Funktionscode)

Wenn der Tastaturpuffer leer ist, wartet die Routine auf eine Eingabe.

TASTATUR INITIALISIEREN [KYINIT] $FFAØ

Einsprung: keine Parameter

Rueckkehr: keine Parameter

Register: keine

K(EY)-FUNKTIONS-SPEICHER INITIALISIEREN [KYSTST] $FF22

Einsprung: (X) Adresse des Datenfeldes mit Charakter-Codes
letzter Code ist $FF
(B) Laenge des Charakter-Feldes (Ø:loeschen)

Rueckkehr: (C) Ø:normal 1:Fehler

DIE ROUTINE 'SNSCOM'

Mit dieser Routine kann man eine Fülle von Kommandos an die SLAVE-CPU übertragen. Dies geschieht folgendermaßen:

Akku A mit 1.M (immer Kommandocode) laden (z.B. $33 für Piepton)
SNSCOM aufrufen durch "JSR $FF19"
Akku A enthält die Antwort der SLAVE-CPU (1.S)
Wenn nötig Akku A mit 2.M laden
SNSCOM aufrufen wie oben
Akku A enthält die Antwort der SLAVE-CPU (2.S)
.....

Um Mißverständnissen durch Übersetzungsfehler vorzubeugen, gebe ich hier den englischen Originaltext wieder.

Ø. SYSTEM COMMANDS

'ACK' = $Ø1

$Ø0: READY CHECK
1.M $Ø0 1.S ACK
$Ø1: SET INITIAL STATUS
1.M $Ø1 1.S ACK
CONSTANTS WHICH ARE INITIALIZED
A: CLEAR BUFFER, SLAVE BUFFER POINTER
B: RECIVED COMMAND = NONE
C: GENERATING POLYNOMIAL $8408
D: BCC REGISTER Ø
E: RS232C BIT RATE 300 BPS
F: RS232C BIT LENGTH 7
G: RS232C PARITY EVEN, STOP BITS 1, CHECK CD
H: CLEAR RS232C STATUS REGISTER
I: SUPER VISOR COMMAND (MASK)
J: EXTERNAL CASSETTE LONG BIAS $139 (2ØØØ BAUD)
K: EXTERNAL CASSETTE SHORT BIAS $9D (1ØØØ BAUD)
L: EXTERNAL CASSETTE BOUNDARY VALUE $E3

```
        M: EXTERNAL CASSETTE GAP BYTES 125
        N: INTERNAL CASSETTE LONG BIAS LOW PULSE TIME $D0
        O: INTERNAL CASSETTE LONG BIAS HIGH PULSE TIME $68
        P: INTERNAL CASSETTE SHORT BIAS $4E
        Q: INTERNAL CASSETTE BOUNDARY VALUE $EA
        R: INTERNAL CASSETTE GAP BYTES 250
        S: INTERNAL CASSETTE COUNTER VALUE 0
        T: INTERNAL CASSETTE COUNTER PULSE LOW
        U: TIMER OVERFLOW INITIAL COUNTER 10
$02: RESET SLAVE-CPU (I/O HOT START INITIALIZE)
     1.M  $02                  1.S  ACK
$03: OPEN SUPER VISOR MASK
     1.M  $03                  1.S  ACK
     2.M  'CODE' =$AA          2.S  "
$04: CLOSE SUPER VISOR MASK
     1.M  $04                  1.S  ACK
$05: READ CONTENTS OF MEMORY
     1.M  $05                  1.S  ACK
     2.M  ADDRESS (HIGH)       2.S  "
     3.M  ADDRESS (LOW)        3.S  DATA
$06: STORE TO MEMORY
     1.M  $06                  1.S  ACK
     2.M  ADDRESS (HIGH)       2.S  "
     3.M  ADDRESS (LOW)        3.S  "
     4.M  DATA                 4.S  "
$07: OR TO MEMORY
     1.M  $07                  1.S  ACK
     2.M  ADDRESS (HIGH)       2.S  "
     3.M  ADDRESS (LOW)        3.S  "
     4.M  DATA                 4.S  "
$08: AND TO MEMORY
     1.M  $08                  1.S  ACK
     2.M  ADDRESS (HIGH)       2.S  "
     3.M  ADDRESS (LOW)        3.S  "
     4.M  DATA                 4.S  "
$09: BARCODE READER POWER ON
     1.M  $09                  1.S  ACK
$0A: BARCODE READER POWER OFF
     1.M  $0A                  1.S  ACK
$0B: JUMP TO LOCATION
     1.M  $0B                  1.S  ACK
     2.M  ADDRESS (HIGH)       2.S  "
     3.M  ADDRESS (LOW)        3.S  "
$0C: BREAK
     1.M  $0C                  1.S  BREAK ACK = $02
$0D: POWER OFF
     1.M  $0D                  1.S  ACK
     2.M  $AA
$0E: SELF CHECK  (DELETED)
     1.M  $0E                  1.S  OR OF BCC REGISTER (0=OK)

NOTE: SUPER VISOR COMMAND = $05, $06, $07, $08, $0B, $0D
NOTE: COMMAND $09 AND $0A DO NOT WORK IN VERSION 1. IN VERSON 2 WORK

1. PRINTER COMMANDS

$10: PRINT BY GRAPHIC PATTERN
     1.M  $10                  1.S  ACK
     2.M  DATA (LS 6 BITS)     2.S  PRINTER ACK = $11
```

```
     DATA TYPE:     7    6    5    4    3    2    1    0
                 <NOT USED> <LAST>                  <FIRST DOT>
     REPEAT 24 TIMES FROM 1 TO 2 TO PRINT ONE DOT LINE.

$11: LINE FEED N DOT LINES
     1.M  $11                    1.S  ACK
     2.M  DATA (LINE N)          2.S  PRINTER ACK = $11
$12: FEED ONE DOT LINE (NON BRAKE)
     (TURN ON MICRO PRINTER'S MOTOR 100 M SEC)
     1.M  $12                    1.S  ACK
```

2. EXTERNAL CASSETTE COMMANDS

```
$20: EXTERNAL CASSETTE READY CHECK
     1.M  $20                    1.S  ACK
$21: SET CASSETTE PARAMETER
     1.M  $21                    1.S  ACK
     2.M  LONG BIAS (HIGH)       2.S  EXTERNAL CASSETTE ACK = $21
     3.M  LONG BIAS (LOW)        3.S  "
     4.M  SHORT BIAS (HIGH)      4.S  "
     5.M  SHORT BIAS (LOW)       5.S  "
     6.M  BOUNDARY VALUE (H)     6.S  "
     7.M  BOUNDARY VALUE (L)     7.S  "
     8.M  GAP BYTES (HIGH)       8.S  "
     9.M  GAP BYTES (LOW)        9.S  "
$22: REMOTE ON
     1.M  $22                    1.S  ACK
$23: REMOTE OFF
     1.M  $23                    1.S  ACK
$24: WRITE ONE BLOCK BY EPSON FORMAT
     1.M  $24                    1.S  ACK
     2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
          00:WITH LONG GAP (DEPEND OF VALUE OF 'GAP BYTES',
                            DEFAULT VALUE = 125 BYTES)
          01:WITH SHORT GAP (15 BYTES $FF)
          FF:WITH LEAD TAPE (625 BYTES $FF)
     3.M  BLOCK TRAIL MODE       3.S  EXTERNAL CASSETTE ACK
          00:WITH LONG GAP (DEFAULT 125 BYTES $FF) AND STOP
                           (DEFAULT VALUE = 125 BYTES)
          01:WITH SHORT GAP (15 BYTES $FF) AND NONSTOP
          FF:WITH TRAIL TAPE (625 BYTES $FF) AND STOP
     4.M  BLOCK SIZE (HIGH)      4.S  EXTERNAL CASSETTE ACK
     5.M  BLOCK SIZE (LOW)       5.S  "
     6.M  DATA CODE              6.S  "
     7.M  .
          .
     N.M  LAST CODE              N.S  "
$25: WRITE TRAIL TAPE
     1.M  $25                    1.S  ACK
     2.M  BYTE COUNT (HIGH)      2.S  EXTERNAL CASSETTE ACK
     3.M  BYTE COUNT (LOW)       3.S  "
$26: SEARCH HEADER BLOCK
     1.M  $26                    1.S  ACK
     2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
     3.M  BLOCK TRAIL MODE       3.S  "
     4.M  BLOCK SIZE (HIGH)      4.S  "
     5.M  BLOCK SIZE (LOW)       5.S  "
$27: SEARCH EOF BLOCK
```

```
      1.M  $27                    1.S  ACK
      2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
      3.M  BLOCK TRAIL MODE       3.S  "
      4.M  BLOCK SIZE (HIGH)      4.S  "
      5.M  BLOCK SIZE (LOW)       5.S  "
$28:  READ ONE BLOCK
      1.M  $28                    1.S  ACK
      2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
           (DUMMY)
      3.M  BLOCK TRAIL MODE       3.S  "
           00:STOP AFTER READ (REMOTE OFF)
           01:NONSTOP AFTER READ (REMOTE ON)
      4.M  BLOCK SIZE (HIGH)      4.S  EXTERNAL CASSETTE ACK
      5.M  BLOCK SIZE (LOW)       5.S  "
$29:  READ START ONE CHARACTER READ MODE
      1.M  $29                    1.S  ACK
      2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
      3.M  BLOCK TRAIL MODE       3.S  "
      4.M  MAX CHARACTERS (H)     4.S  "
      5.M  MAX CHARACTERS (L)     5.S  "
$2A:  WRITE START ONE CHARACTER MODE
      1.M  $2A                    1.S  ACK
      2.M  BLOCK LEAD TAPE MODE   2.S  EXTERNAL CASSETTE ACK
      3.M  BLOCK TRAIL MODE       3.S  "
      4.M  MAX CHARACTERS (H)     4.S  "
      5.M  MAX CHARACTERS (L)     5.S  "
$2B:  SET PULSE MODE (NORMAL/REVERSE)
      1.M  $2B                    1.S  ACK
      2.M  0  NORMAL              2.S  EXTERNAL CASSETTE ACK
           4  REVERSE
```

3. SPEAKER COMMANDS

```
$30:  BEEP BY SCALE'AND TIME
      1.M  $30                    1.S  ACK
      2.M  MUSICAL SCALE          2.S  SPEAKER ACK = $31
           SCALE  0:PAUSE
                  1:DO  2:RE  3:MI  4:FA ...
           FROM 0 TO 57 (DECIMAL)
      3.M  TIME (1 = 0.1 SEC)     3.S  SPEAKER ACK
$31:  BEEP BY FREQUENCY AND TIME (DELETED)
      1.M  $31                    1.S  ACK
      2.M  FREQUENCY (HIGH)       2.S  SPEAKER ACK
      3.M  FREQUENCY (LOW)        3.S  "
           FREQUENCY: TIME OF 1/2 CYCLE (1=1.6 MICRO SEC)
      4.M  TIME (HIGH)            4.S  SPEAKER ACK
      5.M  TIME (LOW)             5.S  "
           TIME: 1=400 MICRO SEC
$32:  BEEP FOR KEY ACCEPT (BEEP 0.03 SEC)
      1.M  $32                    1.S  ACK
$33:  BEEP FOR CTRL/G (BEEP 1 SEC)
      1.M  $33                    1.S  ACK
$34:  SET MELODY DATA
      1.M  $34                    1.S  ACK
      2.M  SCALE (AS 'BEEP')      2.S  SPEAKER ACK
      3.M  TIME (AS 'BEEP')       3.S  "
      4.M    SAME AS '2'
      5.M    SAME AS '3'
```

```
      .
      .
      2N.M   SAME AS '2'
      2N+1.M SAME AS '3'
      LAST.M $FF              LAST.S  SPEAKER ACK
        2N < 48
      NOTE: STORED MELODY DATA WILL BE LOST WHEN MICRO PRINTER COMMAND
            $10 AND $11 IS EXECUTED. BECAUSE THESE COMMANDS USE SAME
            MEMORY BUFFER.
$36:  GO MELODY WHICH DATA IS SET BY COMMAND '$34'
      1.M  $36                  1.S  ACK
```

4. RS232C COMMANDS

'ACK' CODE = $41

```
$40:  RS232C POWER ON (DRIVER ON)
      1.M  $40                  1.S  ACK
$41:  RS232C POWER OFF (DRIVER OFF)
      1.M  $41                  1.S  ACK
$42:  RS232C MODE SET
      1.M  $42                  1.S  ACK
      2.M  BIT RATE (HIGH)      2.S  RS232C ACK
      3.M  BIT RATE (LOW)       3.S  "
      4.M  BIT LENGTH           4.S  "
      5.M  MODE                 5.S  "
           MODE:  7    6    5    4    3    2    1    0
                 <PARITY>  CTS  DSR  RTS  CD  <STOP BITS>
                 PARITY:              00=EVEN  01=ODD  10=NONE
                 CD (CARRIER DETECT): 0=CHECK  1=NO CHECK
                 DSR:                 0=CHECK  1=IGNORED
                 CTS:                 0=CHECK  1=IGNORED
                 STOP BITS:           01=1  10=2  11=3
                 RTS:                 0=LOW    1=HIGH
$43:  READ RS232C STATUS REGISTER
      1.M  $43                  1.S  VALUE OF STATUS REGISTER
$44:  CLEAR STATUS REGISTER
      1.M  $44                  1.S  ACK
$45:  START READ (INTERRUPT MODE)
      AFTER ACCEPT THIS COMMAND, SLAVE-CPU SENDS RS232C RECEIVED DATA
      TO MAIN-CPU VIA SCI (SERIAL COMMUNICATION I/F), AND WHEN ACCEPT
      ANOTHER COMMAND, THIS PROCESS WILL CANCEL.
      1.M  $45                  1.S  ACK
        ENTER RS232C RECEIVE MODE
        FROM SLAVE-CPU TO MAIN-CPU: CONTINUOUS OF RECEIVED DATA
$46:  STOP READ
      1.M  $46                  1.S  ACK
$47:  START READ (READ ONE CHARACTER MODE) (DELETED)
      1.M  $47                  1.S  ACK
        IN READ ONE CHARACTER MODE, RECEIVED CHARS ARE PUSHED INTO
        THE STACK, WHEN ACCEPT READ CHAR COMMAND, ONE CHARACTER IS
        SEND TO MAIN-CPU.
$48:  SET GENERATING POLYNOMIAL
      1.M  $48                  1.S  ACK
      2.M  POLYNONIAL (HIGH)    2.S  RS232C ACK
      3.M  POLYNOMIAL (LOW)     3.S  "
$49:  SET BCC REGISTER
      1.M  $49                  1.S  ACK
      2.M  VALUE (HIGH)         2.S  RS232C ACK
```

```
      3.M  VALUE (LOW)              3.S  "
$4A:  READ BCC REGISTER (HIGH)
      1.M  $4A                      1.S  VALUE OF BCC REGISTER (HIGH)
$4B:  READ BCC REGISTER (LOW)
      1.M  $4B                      1.S  VALUE OF BCC REGISTER (LOW)
$4C:  SERIAL DRIVER ON
      1.M  $4C                      1.S  ACK
$4D:  SET RTS (REQUEST TO SEND)
      1.M  $4D                      1.S  ACK
      2.M  0 (LOW) / 1 (HIGH)       2.S  RS232C ACK
$4E:  READ RS232C BUFFER STATUS IN READ ONE CHARACTER MODE
      1.M  $4E                      1.S  STATUS
$4F:  READ ONE CHARACTER IN READ ONE CHARACTER MODE
      1.M  $4F                      1.S  RECEIVED CHARACTER
        WE CAN USE THE COMMAND AT BAUD FROM 110 TO 1200.

      NOTE: COMMAND $47, $4E AND $4F ARE DELETED (IN VERSION 1. IN
            VERSION 2)

      STATUS REGISTER:    (1:ERROR)
            BIT 0: CARRIER DETECT ERROR
            BIT 1: PARITY ERROR
            BIT 2: OVER RUN ERROR
            BIT 3:
            BIT 7: RECEIVED CHARACTER
```

5. PLUG IN OPTION COMMANDS

```
$50:  CHECK OPTION DEVICE
      1.M  $50                      1.S  OPTIONS CODE
                                         CODE  BIT2-BIT7 : 0
                                         BIT0 : VALUE OF P46
                                         BIT1 : VALUE OF P20
$51:  POWER ON ROM CASSETTE
      1.M  $51                      1.S  ACK
$52:  POWER OFF ROM CASSETTE
      1.M  $52                      1.S  ACK
```

6. INTERNAL CASSETTE COMMANDS

```
$60:  INTERNAL CASSETTE READY CHECK
      1.M  $60                      1.S  ACK
$61:  SET CASSETTE PARAMETER
      1.M  $61                      1.S  ACK
      2.M  LONG BIAS LOW PULSE (HIGH)   2.S  INTERNAL CASSETTE ACK = $61
      3.M  LONG BIAS LOW PULSE (LOW)    3.S  "
      4.M  LONG BIAS HIGH PULSE (HIGH)  4.S  "
      5.M  LONG BIAS HIGH PULSE (LOW)   5.S  "
      6.M  SHORT BIAS (HIGH)            4.S  "
      7.M  SHORT BIAS (LOW)             5.S  "
      8.M  BOUNDARY VALUE (HIGH)        6.S  "
      9.M  BOUNDARY VALUE (LOW)         7.S  "
$62:  GAP BYTE LENGTH
      1.M  $62                      1.S  ACK
      2.M  GAP BYTE COUNT (H)       2.S  INTERNAL CASSETTE ACK
      3.M  GAP BYTE COUNT (L)       3.S  "
$63:  READ SKIP N BYTES
      1.M  $63                      1.S  ACK
```

```
      2.M  SKIP BYTE COUNT (H)  2.M  INTERNAL CASSETTE ACK
      3.M  SKIP BYTE COUNT (L)  3.M  "
$64:  WRITE ONE BLOCK BY EPSON FORMAT
      1.M  $64                  1.S  ACK
      2.M  BLOCK LEAD TAPE MODE 2.S  INTERNAL CASSETTE ACK
           00:WITH LONG GAP (DEPEND OF VALUE OF 'GAP BYTES',
                             DEFAULT VALUE = 125 BYTES)
           01:WITH SHORT GAP (15 BYTES $FF)
           FF:WITH LEAD TAPE (625 BYTES $FF)
      3.M  BLOCK TRAIL MODE     3.S  INTERNAL CASSETTE ACK
           00:WITH LONG GAP (DEPEND OF VALUE OF 'GAP BYTES')
                            (DEFAULT VALUE = 125 BYTES) AND STOP
           01:WITH SHORT GAP (15 BYTES $FF) AND NONSTOP
           FF:WITH TRAIL TAPE (625 BYTES $FF) AND STOP
      4.M  BLOCK SIZE (HIGH)    4.S  INTERNAL CASSETTE ACK
      5.M  BLOCK SIZE (LOW)     5.S  "
      6.M  DATA CODE            6.S  "
      7.M  .
           .
      N.M  LAST CODE            N.S  "
$65:  WRITE TRAIL TAPE
      1.M  $65                  1.S  ACK
      2.M  BYTE COUNT (HIGH)    2.S  INTERNAL CASSETTE ACK
      3.M  BYTE COUNT (LOW)     3.S  "
$66:  SEARCH HEADER BLOCK
      1.M  $66                  1.S  ACK
      2.M  BLOCK LEAD TAPE MODE 2.S  INTERNAL CASSETTE ACK
           (DUMMY)
      3.M  BLOCK TRAIL MODE     3.S  "
           00:STOP AFTER READ
           01:NONSTOP AFTER READ
      4.M  BLOCK SIZE (HIGH)    4.S  "
      5.M  BLOCK SIZE (LOW)     5.S  "
$67:  SEARCH EOF BLOCK
      1.M  $67                  1.S  ACK
      2.M  BLOCK LEAD TAPE MODE 2.S  INTERNAL CASSETTE ACK
           (DUMMY)
      3.M  BLOCK TRAIL MODE     3.S  "
           00:STOP AFTER READ
           01:NONSTOP AFTER READ
      4.M  BLOCK SIZE (HIGH)    4.S  "
      5.M  BLOCK SIZE (LOW)     5.S  "
$68:  READ ONE BLOCK
      1.M  $68                  1.S  ACK
      2.M  BLOCK LEAD TAPE MODE 2.S  INTERNAL CASSETTE ACK
           (DUMMY)
      3.M  BLOCK TRAIL MODE     3.S  "
           00:STOP AFTER READ
           01:NONSTOP AFTER READ
      4.M  BLOCK SIZE (HIGH)    4.S  "
      5.M  BLOCK SIZE (LOW)     5.S  "
$69:  READ START ONE CHARACTER READ MODE
      1.M  $69                  1.S  ACK
      2.M  BLOCK LEAD TAPE MODE 2.S  INTERNAL CASSETTE ACK
           (DUMMY)
      3.M  BLOCK TRAIL MODE     3.S  "
           00:STOP AFTER READ
           01:NON STOP AFTER READ
      4.M  MAX CHARACTERS (H)   4.S  "
```

```
      5.M  MAX CHARACTERS (L)    5.S  "
$6A:  WRITE START ONE CHARACTER MODE
      1.M  $6A                   1.S  ACK
      2.M  BLOCK LEAD TAPE MODE  2.S  INTERNAL CASSETTE ACK
      3.M  BLOCK TRAIL MODE      3.S  "
      4.M  MAX CHARACTERS (H)    4.S  "
      5.M  MAX CHARACTERS (L)    5.S  "
$6B:  STOP EPSON FORMAT WRITE
      1.M  $6B                   1.S  ACK
$6C:  STOP READ ONE CHARACTER MODE
      1.M  $6C                   1.S  ACK
$6D:  SET COUNTER VALUE
      1.M  $6D                   1.S  ACK
      2.M  COUNTER VALUE (HIGH)  2.S  EXTERNAL CASSETTE ACK
      3.M  COUNTER VALUE (LOW)   3.S  "
$6E:  READ COUNTER VALUE (HIGH)
      1.M  $6E                   1.S  COUNTER VALUE (HIGH)
$6F:  READ COUNTER VALUE (LOW)
      1.M  $6F                   1.S  COUNTER VALUE (LOW)
$70:  CHECK IF WRITE ENABLE
      1.M  $70                   1.S  0:ENABLE  FF:DISABLE
$71:  REWIND N COUNT
      1.M  $71                   1.S  ACK
      2.M  COUNT VALUE (HIGH)    2.S  EXTERNAL CASSETTE ACK
      3.M  COUNT VALUE (LOW)     3.S  "
$72:  FEED N COUNT
      1.M  $72                   1.S  ACK
      2.M  COUNT VALUE (HIGH)    2.S  EXTERNAL CASSETTE ACK
      3.M  COUNT VALUE (LOW)     3.S  "
$73:  REWIND TO TOP OF FILE
      1.M  $73                   1.S  ACK
$74:  READ VALUE OF STATUS REGISTER
      1.M  $74                   1.S  VALUE OF STATUS REGISTER
$75:  CLEAR STATUS REGISTER
      1.M  $75                   1.S  ACK
$76:  HEAD LOAD
      1.M  $76                   1.S  ACK
$77:  HEAD UNLOAD
      1.M  $77                   1.S  ACK
$78:  GO REWIND
      1.M  $78                   1.S  ACK
$79:  GO FAST FEED
      1.M  $79                   1.S  ACK
$7A:  GO SLOW FEED
      1.M  $7A                   1.S  ACK
$7B:  STOP (REWIND, FAST FORWARD, SLOW FORWARD)
      1.M  $7B                   1.S  ACK
$7C:  INTERNAL CASSETTE POWER ON AND READ COUNTER PULSE
      1.M  $7C                   1.S  COUNTER PULSE STATUS
                                      (COUNTER POS. $00 OR $80)
$7D:  INTERNAL CASSETTE POWER OFF
      1.M  $7D                   1.S  ACK
```

7. PORT COMMANDS

```
$80:  CONNECT SLAVE-CPU'S PORT XXXX TO MAIN-CPU'S PORT P12 (SFLAG).
      AFTER ACCEPTED THIS COMMAND, VALUE OF DESTINATED ADDRESS OF
      SLAVE-CPU IS STORED TO P34 (CONNECTED TO MAIN-CPU'S P12)
```

```
     CONTINUOUSLY, THEN MAIN-CPU CAN SEE SLAVE PORT'S DATA TO SEE
     P12. (SLAVE P34 IS THE OUT PORT, MAIN P12 THE IN PORT)

     1.M  $80                      1.S  ACK
     2.M  PORT ADDRESS (HIGH)      2.S  "
     3.M  PORT ADDRESS (LOW)       3.S  "
     4.M  PORT BIT                 4.S  "
          (TARGET BIT = 1)

$81: CONNECT 'PLUG-1' TO PORT XX
     AFTER ACCEPTED THIS COMMAND, VALUE OF P40 OF SLAVE-CPU IS
     STORED TO THE DESTINATED PORT OF SLAVE-CPU CONTINUOUSLY.
     P267 (ADDRESS $26 BIT 7, OUT PORT) OF MAIN CPU IS CONN. TO
     THE P40 (IN PORT) OF SLAVE-CPU.

     1.M  $81                      1.S  ACK
     2.M  PORT ADDRESS (HIGH)      2.S  "
     3.M  PORT ADDRESS (LOW)       3.S  "
     4.M  PORT BIT                 4.S  "
          (TARGET BIT = 1)

$82: CONNECT 'SEND' TO PORT XXX  (DELETED)
     1.M  $82                      1.S  ACK
     2.M  PORT ADDRESS (HIGH)      2.S  "
     3.M  PORT ADDRESS (LOW)       3.S  "
     4.M  PORT BIT                 4.S  "
          (TARGET BIT =1)

$83: CONNECT 'PLUG-1' AND 'SEND' TO PORT XXX  (DELETED)
     1.M  $83                      1.S  ACK
     2.M  PORT ADDRESS (H) (PL)    2.S  "
     3.M  PORT ADDRESS (L) (PL)    3.S  "
     4.M  PORT BIT         (PL)    4.S  "
          (TARGET BIT = 1)
     5.M  PORT ADDRESS (H) (SE)    5.S  "
     6.M  PORT ADDRESS (L) (SE)    6.S  "
     7.M  PORT BIT         (SE)    7.S  "
          (TARGET BIT = 1)

NOTE:  COMMAND $82 AND $83 ARE ERASED. (IN VERSION 1, IN VERSION 2)
NOTE:  AFTER THESE COMMANDS, VALUE OF 'PLUG-1' OR 'SEND' DATA IS
       STORED TO DESTINATION ADDRESS. IN THESE MODE, THE COMMAND WILL
       WORK TILL RECEIVE NEXT COMMAND.
```

V A R I A B L E N - S P E I C H E R U N G

EINFACHE VARIABLEN

EINFACHE VARIABLEN WERDEN DIREKT HINTER DEM BASIC-PROGRAMM GESPEICHERT, DER POINTER AUF DEN BEGINN DER EINFACHEN VARIABLEN STEHT IN DEN ADRESSEN 158/159 ($009E/009F). DIE VARIABLEN WERDEN IN DER REIHENFOLGE IHRER ZUORDNUNG GESPEICHERT.

ALLGEMEINES

Die Variablen enthalten im ersten Halb-Byte die Kennung des Variablentyps und im 2. Halb-Byte die Länge des Namens minus 1 ($0-F). Danach kommt der Variablenname, je ein ASCII-Zeichen pro Buchstabe oder Zahl, dann der Inhalt.

KENNUNGEN:
- $2x Integer-Variable
- $3x String-Variable
- $4x Fliesspunkt-Variable (einfache Genauigkeit)
- $8x Fliesspunkt-Variable (doppelte Genauigkeit)

STRING-VARIABLEN

Eine Sonderstellung nehmen die String-Variablen ein, ihr "Inhalt" besteht aus 3 Bytes mit folgender Bedeutung:

1. Byte Stringlänge (0-255 / $00-FF)
2. Byte High Byte Adresse
3. Byte Low Byte Adresse

Die Adresse gibt an, ab welchem Speicherplatz der String gespeichert ist. Dies kann entweder direkt im BASIC-Text sein, wenn eine einfache Zuweisung verwendet wurde, z.B. 'A$="TEXT"', oder im String-Bereich (durch CLEAR definiert), wenn Manipulationen vorgenommen wurden, z.B. 'B$=A$+"VARIABLE"'.

BEISPIEL: A$="ABCD" 30 | 41 | 04 | 3D 22

INTEGER-VARIABLEN

Bei Ganzzahlvariablen besteht der Inhalt aus 2 Bytes (High Byte / Low Byte) zur Speicherung der vorzeichenbehafteten Integer-Zahl (-32768 = $8000 ... +32767 = $7FFF).

BEISPIEL: ABC%=4 22 | 41 42 43 | 00 04

FLIESSPUNKT-VARIABLEN

Variablen mit einfacher Genauigkeit enthalten nach dem Variablennamen 4 Bytes, solche mit doppelter Genauigkeit 8 Bytes.

Die Speicherung ist etwas anders, als im Fliesspunkt-Akkumulator: Das erste Byte (Exponent) ist identisch, das letzte Byte (Vorzeichen) entfällt, dafür enthält das erste Bit des zweiten Bytes das Vorzeichen (plus ist 0, minus ist 1).

BEISPIELE:
- A=4 40 | 41 | 83 00 00 00
- AB#=-.25 81 | 41 42 | 7F 80 00 00 00 00 00 00

ARRAYS (FELDVARIABLEN)

Arrays werden direkt hinter den einfachen Variablen gespeichert, der Pointer auf den Beginn der Arrays steht in den Speicherstellen 160/161 ($00A0/00A1).

Jedes Feld bekommt einen "Header" mit Namen und Dimensionierung. Typkennung und Namensspeicherung ist die selbe, wie bei einfachen Variablen, ebenso die Abspeicherung der Feldelement-Inhalte.

```
BEISPIELE:   DIMAB$(5,6)     31 | 41 42 | 00 85 02 00 07 00 06 04 | ...
             DIMA%(2,3)      20 | 41 | 00 1F 02 00 04 00 03 | ...
             DIMA(1,2,3)     40 | 41 | 00 69 03 00 04 00 03 00 02 | ...
             DIMABC#(4,5)    82 | 41 42 43 | 00 F7 02 06 00 05 | ...
```

Die Pünktchen (...) stehen für die einzelnen Elemente, angefangen bei Element (0) bzw. (0,0), (0,0,0), etc.

VARPTR (VARIABLEN-POINTER)

Diese BASIC-Funktion (Syntaxbeispiel: 'PRINT VARPTR(A)') zeigt auf den Speicherplatz der betreffenden Variable (auf das Byte direkt hinter dem Variablennamen, also das erste Byte des "Inhalts"). Sie ist bei allen Variablentypen (auch Arrays) anwendbar.

Bei Stringvariablen wird, wie bereits beschrieben, hier nur der Platz gespeichert, an dem der eigendliche String steht. Hier bekommt man die Stringlänge durch:

```
?PEEK(VARPTR(A$))
```

und die Speicherstelle, wo der String beginnt durch:

```
?PEEK(VARPTR(A$)+1)*256+PEEK(VARPTR(A$)+2)
```

Durch "verbiegen" des Pointers kann man ihn auf einen beliebigen Bereich richten und den Inhalt einfach als normale Stringvariable auslesen. Um in diesen Bereich poken zu können, muss man erst 'POKE126,128' geben. Von Maschinenprogrammen aus ist dies immer möglich.

Der Pointer in den Adressen 162/163 ($00A2/00A3) zeigt auf das Ende des Variablenbereichs.

Kontakte

Autorisierte Händler in Ihrer Nähe

Wenn Sie Hard- oder Software-Probleme, Fragen oder Wünsche haben, dann versuchen Sie es doch zuerst einmal beim nächstgelegenen Epson-Händler. Hier finden Sie Adressen aus dem deutschsprachigen Raum und von den wichtigsten Distributoren in Europa, soweit sie bis April 1984 von Epson autorisiert wurden.

Postleitzahlgebiet 1000

Horn & Görwitz GmbH & Co.
An der Urania 16–18
1000 Berlin 30
Telefon 030/21091
Telex: 184937

Computer Shop Minhoff & Co. GmbH
Bundesallee 160
1000 Berlin 31
Telefon: 030/8545055
Telex: 181832

Computer Shop
Minhoff & Co. GmbH
Hohenzollerndamm 47a
1000 Berlin 31
Telefon: 030/860642

TCV-Stroetmann/
Text-Computer-Vertriebs GmbH
Kurfürstenstr. 72–74
1000 Berlin 30
Telefon: 030/2621485

Runow Bürcelektronik
Keithstr. 26
1000 Berlin 30
Telefon: 030/2611126

Postleitzahlgebiet 2000

Richard Kiessling Ing.
Kleine Reichenstr. 1
2000 Hamburg 11
Telefon: 040/33409-0
Telex: 402011

Computer Partner
Rothenbaumchaussee 83
2000 Hamburg 13
Telefon: 040/2280094 + 448351

HABA-Technik
Münsterstraße 9
2000 Hamburg 54
Teiefon: 040/566011

Walter Kluxen GmbH
Nordkanalstr. 52
2000 Hamburg 1
Telefon: 040/237010
Telex: 2162074

Radix Bürotechnik
Gritzka + Militzer
Bornstraße 4
2000 Hamburg 13
Telefon: 040/441695

Runow Büroelektronik GmbH
Bachstr. 104
2000 Hamburg 76
Telefon: 040/2201155

Createam Microcomputer GmbH
Bramfelder Chaussee 300
2000 Hamburg 71
Telefon: 040/6416473

Teletex & Computer
Vertriebs GmbH
Billbrookdeich 34
2000 Hamburg 74
Telefon: 040/7331063

Hartwig Ohnimus
Am Sande 16
2120 Lüneburg
Telefon: 04131/401910

Computer-Shop Sienknecht
Lünertorstr. 16
2120 Lüneburg
Telefon: 04131/128146

Bürotechnik Nord GmbH
Norderstraße 22
2250 Husum
Telefon: 04841/2455

Carl Reese GmbH & Co.
Rendsburger Landstr. 196–198
2300 Kiel 1
Telefon: 0431/6891-1
Telex: 292981

micro-computer-schütte gmbh
Rendsburger Str. 22–24
2350 Neumünster
Telefon: 04321/14001 + 2

Jessen & Lenz
Wahmstr. 36
2400 Lübeck 1
Telefon: 0451/705030/705151

Manfred Schoeffel
Ingenieurbüro
Rohlsdorfer Weg 33
2409 Techau
Telefon: 04504/281

Paul F. W. Meister GmbH & Co.
Hastedter Dorfstr. 20
2800 Bremen 1
Telefon: 0421/498055
Telex: 246215

Weber-Funk GmbH
Emil-von-Behring-Str. 6
2800 Bremen 34
Telefon: 0421/490010/19
Telex: 244952

Hans Schröder
Computer-Systeme GmbH
Föhrenstraße 19
2800 Bremen 1
Telefon: 0421/459779

Dörrbecker & Co.
Handelsges. mbH
Sögestraße 36–38
2800 Bremen 1
Telefon: 0421/14955

MBC-Modulare
Büro-Computer
Handelsges. mbH
Woltmershauser Str. 161/163
2800 Bremen 1
Telefon: 0421/541294

Prisma
Computer-Elektronik GmbH
Heinrichstraße 26
2807 Achim
Telefon: 04202/82954

Münch Datensysteme
Brinkstr. 43
2842 Lohne
Telefon: 04442/2516

Multidata
Jägerstr. 64–66
2900 Oldenburg
Telefon: 0441/73024
Telex: 25786

Renken Computer Systeme
Bismarckstr. 150
2940 Wilhelmshaven
Telefon: 04421/12636

Bürohaus Gerhard Duin oHG
Große Straße 6
2970 Emden 1
Telefon: 04921/20502

Postleitzahlgebiet 3000

TCV-Stroetmann
Nikolaistr./
Ecke Nordfelder Reihe 27
3000 Hannover 1
Telefon: 0511/14658-59
Telex: 175118433

trendDATA Computer GmbH
Am Marstall 20/22
3000 Hannover 1
Telefon: 0511/14629

ASS Microcomputer GmbH
Goethepiatz 10
3000 Hannover 1
Telefon: 0511/323625

TIS
Text-Informations-Systeme GmbH
Adelheidstraße 3
3000 Hannover 1
Telefon: 0511/817025

Ludwig Haupt jr.
Gerhard-Kamm-Str. 2
3100 Celle-Westercelle
Telefon: 05141/83045

Datentechnik
W. Werner + G. v. Juroszek
Torstr. 4
3170 Gifhorn
Telefon: 05371/56058

Witte Bürotechnik
Bäckerstr. 22
3250 Hameln
Telefon: 05151/7595 + 7596

Apel Büro-Center GmbH
Heinrich-Büssing-Ring 40
3300 Braunschweig
Telefon: 0531/791001
Telex: 175318130
Teletex: 5318130

Kneisner Elektronik
Senefelderstraße 16
3300 Braunschweig
Telefon: 0531/613552

KUPA Datenservice GmbH
Am Scheunenkamp 16
3307 Königslutter
Telefon: 05353/7724

Computerdienst
Brockhaus & Müller GmbH
Weender Landstr. 3
3400 Göttingen
Telefon: 0551/45977

ECG Computer
Enkephalon Computer GmbH
Rudolf-Winkel-Straße 5a
3400 Göttingen
Telefon: 0551/68875

Retron GmbH
Rodeweg 18
3400 Göttingen
Telefon: 0551/9040

Wilh. F. Kassebeer
GmbH & Co. KG
Büro-Musterhaus
Matthias-Grünewald-Str. 42
3410 Northeim
Telefon: 05551/2097

K & R
Computer Shop GmbH
Marktstraße 29
3440 Eschwege
Telefon: 05651/6772

Bernhard Starke
Kohlenstr. 49–51
3500 Kassel
Telefon: 0561/2991
Telex: 992505

Hermann Fischer oHG
Rudolf-Schwander-Str. 5
3500 Kassel
Telefon: 0561/770087

Postleitzahlgebiet 4000

Datec
Software GmbH
Roßstraße 13
4000 Düsseldorf 30
Telefon: 0211/482085-86

Data Becker GmbH
Merowingerstr. 30
4000 Düsseldorf 1
Telefon: 0211/310010
Telex: 8582874

Helmut Rennen GmbH & Co. KG
Martinstr. 55
4000 Düsseldorf 1
Telefon: 0211/306098

Dr. Ing. Quaye GmbH
Jupiterstr. 28
4044 Kaarst 1
Telefon: 02101/604021-23
Telex: 8517626

SYMIC Computer Vertriebs GmbH
Lessingstraße 1a
4050 Mönchengladbach 1
Telefon: 02161/200865
+ 02161/18751

Kirchner Elektronik
Grabenstr. 90
4100 Duisburg-Neudorf
Telefon: 0203/372128
Telex: 855531

Helmut Rennen GmbH & Co. KG
Mercatorstr. 80
4100 Duisburg 1
Telefon: 0203/24926

S.V.I. Computercentrale
Carl-Wilhelm-Str. 27
4150 Krefeld
Telefon: 02151/64285

Feldmann & Luft
Emmericher Str. 223
4190 Kleve
Telefon: 02821/91038
Telex: 811797

micro Computer GmbH
Münsterstraße 62–64
4290 Bocholt
Telefon: 02871/7099

Helmut Rennen GmbH
Altendorfer Str. 40
4300 Essen
Telefon: 0201/237139

Computersysteme
Michael Miecko
Juliusstraße 15–17
4300 Essen 1
Telefon: 0201/232428

Computer Centrale
Dortmunder Str./Ecke Douaistr. 1
4350 Recklinghausen
Telefon: 02361/45708

Computer Stümpel
Hammer Straße 223
4400 Münster
Telefon: 0251/796929 + 20

GAO Computerhaus
Aegidiimarkt 5
4400 Münster
Telefon: 0251/706-1
Telex: 892852

Füssner Computersysteme GmbH
Hörstkamp 7
4440 Rheine 1
Telefon: 05971/1253-9

Scherz & Cramer GmbH
Aversdieckstraße 1
4500 Osnabrück-Hellern
Telefon: 0541/46094
Telex: 94937

Richard Müller GmbH & Co.
Viktoriastr. 14
4600 Dortmund 1
Telefon: 0231/528722
Telex: 17231342

Systemhaus
Reschke GmbH & Co. KG
Hohe Straße 21a
4600 Dortmund
Telefon: 0231/160014

BO-DATA GmbH & Co. KG
Querenburger Höhe 209
4630 Bochum 1
Telefon: 0234/702277
Telex: 234313

Beate Vollrath
Computer und Zubehör
Kirchstr. 28
4650 Gelsenkirchen
Telefon: 0209/209291

REED
Dirk Röhricht
Lütge Bruch 1a
4690 Herne 1
Telefon: 02323/33163

Rüter GmbH & Co. KG
Heinrich-Gustav-
Heinemannstraße 19–21
4700 Hamm 1
Telefon: 02381/21068

IDS-Industrielle Datensysteme
GmbH
Marktstr. 22
4780 Lippstadt
Telefon: 02941/5011

Computer Böhm GmbH
Friedrichstr. 9
4790 Paderborn
Telefon: 05251/23467

A + L DATA GmbH
Kleine Datensysteme
Hauptstr. 81
4800 Bielefeld 14
Telefon: 0521/431524

proCom
Am Stadion 86
4800 Bielefeld 11
Telefon: 05205/5047

Friedr. Ritschel
GmbH & Co. KG
Gehrenberg 2
4900 Herford
Telefon: 05221/53286

Postleitzahlgebiet 5000

Büromaschinen Braun
Am Rudolfplatz GmbH
Habsburgerring 9–13
5000 Köln 1
Telefon: 0221/219171

Gasper Computers GmbH
Quater Markt 5
5000 Köln 1
Telefon: 0221/210597-8
Telex: 8885506

Ultra-Electronic
Helmut Proxa GmbH & Co. KG
Eupener Str. 1
5000 Köln 41
Telefon: 0221/491091
Telex: 8886627

Franz Pütz GmbH & Co. KG
Im Gewerbegebiet 37
5000 Köln 71
Telefon: 0221/5903086

Proxa Computer
Aachener Str. 29
5000 Köln 1
Telefon: 0221/491091

Bürotechnik Runte oHG
K.-Adenauer-Platz 2
5090 Leverkusen 1
Telefon: 0214/42178

City-Computer Leverkusen
Hauptstraße 82
5090 Leverkusen
Telefon: 0214/48962

Franz Pütz GmbH & Co. KG
Adalbertstr. 90
5100 Aachen
Telefon: 0241/22444

Mittelrhein. Rechenzentrum GmbH
Kölner Landstraße 240
5160 Düren
Telefon: 02421/71075

Intro
Rather Str. 33–37
5168 Nideggen
Telefon: 02427/6061-62
Telex: 833711

Franz Pütz GmbH & Co. KG
Südstr. 39–41
5180 Eschweiler
Telefon: 02403/25097
Telex: 832188

Cicero Computer
Mieth & Richarz
Bahnstr. 7
5205 St. Augustin 1
Telefon: 02241/331465
Telex: 889494

ACT-Angewandte
Computer Technik
Langgasse 96
5216 Niederkassel 5
Telefon: 0228/453500

Böttcher Energineering GmbH
Hermannsplatz 8
5243 Herdorf/Sieg
Telefon: 02744/5998
Telex: 876321

Eugen Haas
Gummersbacher Straße 44
5270 Gummersbach-
Niedersessmar
Telefon: 02261/3001-0

Renate Nensel
Am Bismackplatz 2
5270 Gummersbach 1
Telefon: 02261/26433

Gaetan-Data
Ges. f. Datenverarbeitung mbH
Weberstraße 132
5300 Bonn 1
Telefon: 0228/214725

Plasmann Computer Center GmbH
Heerstraße 175–179
5300 Bonn 1
Telefon: 0228/650965
Telex: 885775

ID Computer-Systeme GmbH
Himberger Str. 5a
5340 Bad Honnef 6
Telefon: 02224/80106-126

K-Electronics
Billiger Str. 70
5350 Euskirchen
Telefon: 02251/63078

Görlitz Computerbau
Lambertstr. 49
5400 Koblenz
Telefon: 0261/27500

Microcomputerzentrum
Rhein-Hunsrück
Hauptstraße 2
5448 Spesenroth
Telefon: 06762/8401/00

mikra GmbH
Microcomputeranwendung
Breite Straße 58
5470 Andernach
Telefon: 02632/47666

C. O. S.
Computer Organisation GmbH
Zuckerbergweg 21
5500 Trier
Telefon: 0651/44077

Fritz Lehr
Bürofachgeschäft
Güterstraße 82
5500 Trier
Telefon: 06151/45025

T.E.D.
Computer-Systeme GmbH
Abt-Richard-Str. 16
5568 Daun
Telefon: 06592/657

Röth Datensysteme GmbH & Co. KG
Saarbrücker Str. 40–42
5600 Wuppertal 2
Telefon: 0202/584-1
Telex: 8591316

Röth Datensysteme GmbH & Co. KG
Otto-Hahn-Str. 19
5600 Wuppertal 21
Telefon: 0202/584267

Computerladen Joerss GmbH
Wuppertaler Str. 58
5650 Solingen 1
Telefon: 02122/51637

Johannes Gerlach
Datentechnik GmbH
Zwg.-Niederlassung Carl Meister
Steinhausstr. 36
5800 Hagen
Telefon: 02331/64444,47

Möller Bürotechnik
Friedrich-Kirchhoff-Str. 4
5860 Iserlohn
Telefon: 02371/40022

Büroeinrichtungshaus
LOUIS HEES GmbH
Weidenauer Str. 84
5900 Siegen 21
Telefon: 0271/72602

Postleitzahlgebiet 6000

Kegelmann Computerhaus GmbH
Gaußstr. 12
6000 Frankfurt 1
Telefon: 069/446016

Canberra Elektronik GmbH
Hahnstr. 70
6000 Frankfurt 1
Telefon: 069/6666087

City Computer GmbH
Wittelsbacherallee 153
6000 Frankfurt 60
Telefon: 069/440646

Natic + EDV
Rossertstr. 6
6000 Frankfurt
Telefon: 069/7240368

Computer Systems GmbH
Buchrainstr. 34
6000 Frankfurt 70
Telefon: 069/654343

MAS
Schwarzwaldstraße 122
6000 Frankfurt 71
Telefon: 069/676015

MIDAS
Micro-Datensysteme GmbH
Brönnerstr. 15
6000 Frankfurt 1
Telefon: 069/597022

Mirwald-Elektronik GmbH
Adalbertstraße 15
6000 Frankfurt 90
Telefon: 069/703538

Henneveld KG
In der Zeil 127
6000 Frankfurt 1
Telefon: 069/281482-83

micro Computer-Systeme
Alsfelder Str. 7
6100 Darmstadt
Telefon: 06151/76031

EBG Darmstadt
Lothar Schanuel GmbH
Heidelberger Str. 73
6100 Darmstadt
Telefon: 06151/313890
Telex: 4197160

Walter GmbH
Kopier- und Zeichentechnik
Butzbacher Weg 7
6100 Darmstadt
Telefon: 06151/76061

Softdoor GmbH
Hauptstr. 20
6140 Bensheim
Telefon: 06251/68635

Natic & EDV Vertrieb
Nerotal 73
6200 Wiesbaden
Telefon: 06121/51745

DMS Büroorganisation GmbH
Schwimmbadstr. 36
6301 Heuchelheim
Telefon: 0641/61014–16

Steinweller oHG
Petersbergerstr. 21
6400 Fulda
Telefon: 0661/75051

Electronic Service
Karlstr. 40
6400 Fulda
Telefon: 0661/76681

EMC-Software
Nonnengasse 7–9
6411 Kunzell 1
Telefon: 0661/71354

Ahlbrandt GmbH
Beethovenstraße 3
6420 Lauterbach
Telefon: 06641/3081

Hölt! Electronic GmbH
Badestube 12
6430 Bad Hersfeld
Telefon: 06621/2790
Telex: 493125

CP Computer Products
Hospitalstraße 2
6450 Hanau/Main
Telefon: 06181/22411

CKL Ing.-Büro f.
Comp.-Komplettlösungen
Schulstr. 14
6451 Ronneburg 1
Telefon: 06184/4292

Elphotec Computer Systeme GmbH
Schießgartenstr. 7
6500 Mainz
Telefon: 06131/231947-48

A. Frank Haus der Computer
Rheintalstraße 36
6500 Mainz 43
Telefon: 06131/86061-62

CCS Security
Sicherheitstechnik-Datentechnik
GmbH
Otto-Wels-Str. 58
6500 Mainz-Weisenau
Telefon: 06131/85603

Georg Steinmetz OHG
Neumarkt 4
6520 Worms
Telefon: 06241/6868

Wiko GmbH
Computer Textsysteme
Mainzer Straße 116–118
6600 Saarbrücken
Telefon: 0681/63444

W. N. Pfeiffer
Büromaschinen KG
Beethovenstr. 28
6600 Saarbrücken 3
Telefon: 0681/32711-12

Data Service
Mundenheimer Straße 232
6700 Ludwigshafen
Telefon: 0621/581873

MICOS
Computer Vertriebs GmbH
Landauerstr. 36
6720 Speyer
Telefon: 06232/28497

MKV
Microcomputer Vertriebsges. mbH
Landauer Str. 47
6720 Speyer
Telefon: 06232/77216

Data Service GmbH
Kramstr. 21–23
6740 Landau/Pfalz
Telefon: 06341/83072–73

Data Service GmbH
Königstr. 20–22
6750 Kaiserslautern
Telefon: 0631/16081-2

Kirch Computersysteme
Dr. Rudolf-Breitscheidstr. 16
6750 Kaiserslautern
Telefon: 0631/65333

Josef Arzt KG
Textverarbeitung
N3, 10/Kunststraße
6800 Mannheim 1
Telefon: 0621/23844
Telex: 463539

Postleitzahlgebiet 7000

Haller & Tietze GmbH
Leuschnerstr. 16
7000 Stuttgart 1
Telefon: 0711/293792

Kübler
Büro- und Datentechnik
Silberburgstr. 145
7000 Stuttgart 1
Telefon: 0711/610651
Telex: 722758

Fritz Computer GmbH
Schulze-Delitzsch-Str. 22
7000 Stuttgart 80
Telefon: 0711/7800238-39

Dontenwill
Kronprinzstraße 6
7000 Stuttgart 6
Telefon: 0711/294665

Schreiber Elektronik GmbH
Alte Poststr. 2
7000 Stuttgart 1
Telefon: 0711/227099

Fischer Software GmbH
Friesenstr. 33
7000 Stuttgart 40
Telefon: 0711/871066

Spectra
Computersysteme GmbH
Benzstraße 17
7022 Leinfelden
Telefon: 0711/751081
Telex: 7255155

Jo-Microcomputer GmbH
Hintere Schmiedgasse 33
7070 Schwäb. Gmünd
Telefon: 07171/5828

Böhmer Electronic
Wilhelm-Zapf-Str. 9
7080 Aalen
Telefon: 07361/62686
Telex: 713773

SEEL Büromusterhaus GmbH & Co.
Am Wollhaus 6
7100 Heilbronn
Telefon: 07131/68401

Bechtle GmbH EDV-Technik
Wilhelm-Str. 33
7100 Heilbronn
Telefon: 07131/87731

Wilhelm Geiger GmbH
Werder Str. 134
7100 Heilbronn
Telefon: 07131/60785

SEEL Büromusterhaus
GmbH & Co. KG
Benzstr. 4
7129 Ilsfeld
Telefon: 07062/6020
Telex: 728667

Förster + Scharf
Hauptstr. 12
7157 Murrhardt
Telefon: 07192/7061

Sutter
Datensysteme GmbH
Bahnhofstr. 31
7200 Tuttlingen
Telefon: 07461/79491

Bauknecht GmbH
Rottweilerstraße 10
7230 Schramberg-Sulgen
Telefon: 07422/6012

Flaig + Allgaier
Hölzle 58
7233 Lauterbach
Telefon: 07422/22599

Riese Electronic
GmbH & Co. Vertrieb KG
Junghansstr. 16
7240 Horb
Telefon: 07451/2441
Telex: 765326

Rielo Elektronik
Reichenbergerstr. 1
7250 Leonberg
Telefon: 07152/44888
Telex: 7245772

Comput-Elektronik
Gartenstr. 39
7320 Göppingen
Telefon: 07161/70665

Rolf Winckler
Computer- und Funkstudio
Stuttgarter Str. 43
7320 Göppingen
Telefon: 07161/12221

Maier + Partner GmbH
Dieselstraße 6
7410 Reutlingen 11
Telefon: 07121/54025
Telex: 729975

PCB Personal-Computer
Büro System GmbH
Oskar-Kalbfellplatz 8
Nordsternhaus
7410 Reutlingen
Telefon: 07121/270443

MKV Microcomputer
Vertriebsgesellschaft mbH
Rüppurrer Str. 2d
7500 Karlsruhe
Telefon: 0721/373071 und 26915
Telex: 7826311

Baum & Partner
Sophienstr. 89
7500 Karlsruhe
Telefon: 0721/855515

Softronic GmbH
Kaiserpassage 2
7500 Karlsruhe 1
Telefon: 0721/21313
Telex: 7826633

ISYS Computertechnik
Gartenstr. 56b
7500 Karlsruhe
Telefon: 0721/856145

Comtronic GmbH
Akademiestraße 24
7500 Karlsruhe 1
Telefon: 0721/25931 + 29615

MEMA GmbH
Ludwig-Marum-Straße 23
7500 Karlsruhe
Telefon: 0721/551921

Systemberatung
Jürgen Wagner
Blumenstraße 2
7500 Karlsruhe
Telefon: 0721/29484

MICO-ELECTRONIC GmbH
Neue Bahnhofstr. 2
7504 Weingarten
Telefon: 07244/8292
Telex: 7825881

dm-Technik
Durlacher Str. 39
7530 Pforzheim
Telefon: 07231/13939
Telex: 7263602

Büro 2000
Stahr + Beckmann GmbH
Jörg-Ratgeb-Straße 1
7530 Pforzheim
Telefon: 07231/22074-76

Leonhardt Electronic
In der Jeuch 3
7600 Offenburg
Telefon: 0781/75974
Telex: 752918

Fischer Computer Systeme
Gottlieb-Daimler-Str. 7
7730 Villingen
Telefon: 07721/70046

Elektronik Obser
Gartenstr. 27
7750 Konstanz
Telefon: 07531/22929
Telex: 733324

Meisterknecht GmbH
Habsburger Str. 125
7800 Freiburg i. Br.
Telefon: 0761/31666

Dontenwill
Steinbrückstr. 13
7880 Bad Säckingen
Telefon: 07761/3093
Telex: 792339

Kumatronik
Kasernstr. 47
7910 Neu-Ulm
Telefon: 0731/722999

Kumatronik
Markdorfer Str. 27
7980 Ravensburg-Bavendorf
Telefon: 0751/9541 bis 9543
Telex: 732911

Postleitzahlgebiet 8000

Dontenwill GmbH
Landwehrstr. 40
8000 München 2
Telefon: 089/597993

Logitec GmbH
Einsteinstr. 111
8000 München 80
Telefon: 089/472069
Telex: 5213206

Schulz Computer-Shop
Schillerstr. 22
8000 München 2
Telefon: 089/597330

Schulz Bürotechnik GmbH
Dachauer Str. 192
8000 München 50
Telefon: 089/14820
Telex: 529654

Computerfachgeschäft GmbH
Maximilansplatz 22
8000 München 2
Telefon: 089/227301

Heninger Microcomputer GmbH
Landwehrstr. 39
8000 München 2
Telefon: 089/557067

MKV Mikrocomputer
Vertriebsgesellschaft mbH & Co.
Computermarkt KG
Landwehrstr. 75
8000 München 2
Telefon: 089/3105613

AiD Systemhaus GmbH
Belgradstr. 9
8000 München 40
Telefon: 089/3088001

Karl Finkenzeller GmbH
Taunusstr. 31
8000 München 40
Telefon: 089/3871-1
Telex: 529113

Ludwig
Bürotechnik + Computer
Ingolstädter Straße 62L
8000 München 45
Telefon: 089/3113066

K. L. Spieß
Computersysteme
Joseph-Spitalstr. 7/1
8000 München 2
Telefon: 089/2608161

Beste Marketing
Bodenseestraße 17–19
8000 München 60
Telefon: 089/8345097

MC Home Computer
Grilliparzer Straße 31
8000 München 80
Telefon: 089/476888

MMOS
Computer GmbH
Schopenhauerstr. 2B
8014 Neubiberg
Telefon: 089/6015705
Telex: 5216376

Mirwald Electronic GmbH
Fasanenstr. 8b
8025 Unterhaching/München
Telefon: 089/6111224 + 6112040
Telex: 5213476

Münzenloher GmbH
Tölzer Str. 5
8150 Holzkirchen
Telefon: 08024/1814

Messtec
Meichelbeckstr. 2
8174 Benediktbeuren
Telefon: 08857/8591

Georg Mylius
Computersysteme GmbH
Nikolaistraße 10
8200 Rosenheim
Telefon: 08031/34041

Szeredy Elektronik
Am Salzstadel 17
8200 Rosenheim
Telefon: 08031/36625

Dipl.-Ing. Hans Prechtl
Heubergstr. 114
8229 Mitterfelden
Telefon: 08654/50322

Bauer Elektronik Vertriebs GmbH
Mitterweg 5
8300 Ergolding
Telefon: 0871/78179
Telex: 58311

EDV-Kommunikation Zeitlhöfer
Bäckergasse 2
8374 Viechtach
Telefon: 09942/3616

TS Computer-Technik
Raiffeisenstr. 3
8390 Passau
Telefon: 0851/83309

Herbert Taulien
An der Brunnstube 22
8400 Regensburg
Telefon: 0941/35900

Roscher GmbH
Bürozentrum
Donaustauferstr. 185
8400 Regensburg
Telefon: 0941/48066

Glaser + Partner
Glückaufstr. 3
8451 Kümmersbrück
Telefon: 09621/71414

BDS Büro- und Datensysteme
Ettmannsdorfer Str. 8
8460 Schwandorf
Telefon: 09431/20478

FBH Datenverarbeitung GmbH
Brunner-Bey-Str. 35
8473 Pfreimd
Telefon: 09606/1411
Telex: 63919

Roscher
Centrum für Büro + Technik
Liebigstr. 11
8480 Weiden/Oberpfalz
Telefon: 0961/3050
Telex: 63818

Alois Hösl
Unterer Markt 11
8480 Weiden/Opf.
Telefon: 0961/42433

Wagner GmbH
Fürther Str. 338
8500 Nürnberg 1
Telefon: 0911/329060
Telex: 622545

Micropoint-Electronic GmbH
Werderstr. 18a
8500 Nürnberg 20
Telefon: 0911/537440

Wieland-Löw GmbH
Äußere Further Str. 4a
8501 Wendelstein
Telefon: 09129/8015

Büro 2000 Haas
Dresdner Str. 5
und Friedrichstr. 9
8520 Erlangen
Telefon: 09131/12010

Unidata Computersysteme GmbH
Parkstr. 3
8580 Bayreuth
Telefon: 0921/56688

Gerd's Elektroladen
Gerhard Herold
Kulmbacher Str. 20
8580 Bayreuth
Telefon: 0921/64868

Meier + Beil
Max-Reger-Str. 5
8590 Marktredwitz
Telefon: 09231/8556
Telex: 641283

Steinwald Elektronik
Egerstr. 23
8590 Marktredwitz
Telefon: 09231/62971

Stefan Hastreiter
Gartenstr. 12
8591 Thiersheim
Telefon: 09233/8937

BÜRO-ZENTRUM
A. + R. Kutz
Am Kranen 12a
8600 Bamberg
Telefon: 0951/27808

Günter Lapenat
Haidberg 192
8665 Zell/Ofr.
Telefon: 09257/202

ROTRONIK
EDV-Entwicklungen
Bahnhofstraße 13
8672 Selb
Telefon: 09287/3099

Schöll Computer-Center
Dominikanerplatz 5
8700 Würzburg 1
Telefon: 0931/50488
Telex: 68706

Haller GmbH
Büttnerstr. 29
8700 Würzburg
Telefon: 0931/16705

Schöll Computer-Center
Münchner Str. 6
8730 Bad Kissingen
Telefon: 0971/3254

Viktor Willgerodt
GmbH & Co. KG
Hanauer Str. 12
8750 Aschaffenburg
Telefon: 06021/21375

F. Kutzschbach
Am Weiher 100
8862 Ehringen
Telefon: 09081/79226

Bürotechnik Bissinger GmbH
Industriestr. 18
8883 Gundelfingen
Telefon: 09073/7023-25

C + S Computer und Software
GmbH
Klinkerberg 8
8900 Augsburg
Telefon: 0821/515120

Adrian Karle
Ziegelgrundweg 26
8901 Gablingen
Telefon: 08230/1271

Taesler Computer-Vertrieb
Breslauer Str. 53
8902 Neusäss
Telefon: 0821/467705

HSBS Computersysteme
Zangmeisterstr. 3
8940 Memmingen
Telefon: 08331/12059
Telex: 541024

Manfred Schweizer KG
Pfälzer Straße 16
8940 Memmingen
Telefon: 08331/12220

PB-Data GmbH
Kemptener Str. 12
8950 Kaufbeuren
Telefon: 08341/7667

Graf Elektronik Systeme GmbH
Magnusstr. 13
8960 Kempten
Telefon: 0831/6211

Schweiz

Adcomp Equipment AG
Lerzenstr. 27
CH-8953 Dietikon
Telefon: 0041-1-7414111
Telex: 045-58657
(Herr Studer)

Excom AG
Einsiedlerstr. 31
CH-8820 Wädenswil
Telefon: 0041-1-7807414
Telex: 045-875037
(Herr Götti)

Österreich

Hayward Computerperipherie
Alpenstr. 108a
A-5033 Salzburg
Telefon: 0043-662-20026 + 22856
Telex: 047-633336
(Herr Hayward)

Belgien

Manudax S. A.
Emile Bockstaellaan 5
B-1020 Brüssel
Telefon: 0032-2-4250660
Telex: 046-21183
(Herr de Hazelaer)

Griechenland

Electronic Commercial
Services S.A.
Ermou & Fokionos 8
Athen
Telefon: 0030-1-3253839
Telex: 0601-223996
(Herr Dr. Kanellopoulos)

Niederlande

Manudax B.V.
Meerstraat 7
NL-5473 ZG Heeswijk
Telefon: 0031-4139-2901
Telex: 044-74810
(Herr Rakhorst)

Türkei

ROMAR Pazarlama Sanayi ve
Ticaret A.S.
Rihtim Caddesi Nesli Han Kat 4
Karaköy-Istanbul
Telefon: 00901-1-447502
(Herr Türkkan/Herr Karapazar)

So schicke ich Programme ein!

Ihre Programm-Einsendung hat größere Chancen in CHIP-SPECIAL veröffentlicht zu werden, wenn Sie bei der Einsendung folgendes beachten:

1. Ihr Begleitschreiben sollte neben Ihrer Anschrift auch Ihre Telefonnummer enthalten, die Angabe Ihres Geburtstages oder des Lebensalters wäre wünschenswert.
2. Sämtlicher Briefwechsel muß unbedingt in der "Betreffzeile" anzeigen, um welchen Computer es sich handelt.
3. Das eingereichte Computer-Programm sollte in Form eines Datenträgers (Diskette oder Kassette), wenn möglich mit Listing, aber unbedingt mit einer ausführlichen Dokumentation (!!!) eingereicht werden. Jedes Blatt, jeden Datenträger bitte mit eigenem Name und Computertyp (!!!) beschriften.
4. Die Dokumentation zum Programm sollte einen Titelvorschlag enthalten, den Hinweis, welche Erweiterungen oder Peripherie notwendig ist, den Grund, weshalb es sich für den Leser lohnen könnte, gerade dieses Programm abzutippen, eine Beschreibung der Programmfunktionen, eine exakte Programmbeschreibung (mit Variablendefinition und Hinweis auf wesentliche Programmzeilen).
5. Mehrere eingereichte Programme schreiben Sie bitte auf einzelne Blätter, damit wir diese unabhängig von Ihrem Anschreiben auf mehrere unserer Tester aufteilen können. Die Programme selbst können jedoch auf einem einzigen Datenträger abgespeichert werden (bitte mit mindestens einer Sicherheitskopie).
6. Der Umfang eines Programmes ist nicht limitiert. Was Chancen hat, abgetippt zu werden, erscheint in CHIP-Special, was uns zu umfangreich düngt, wird einem Zusatztest für Datenträgerverkauf auf CHIP-SOFT unterzogen. Auch hier ist eine ausreichende Programmbeschreibung wichtig. Auch wenn sich das Programm x-mal im Listing selbst erklärt, ohne Programmbeschreibung können wir mit Ihrem Programm nichts anfangen.
7. Welche Themen sind gefragt? Alle Themen, die für größere Interessentenkreise in Frage kommen. Gefragt sind "aktive" Programme für Freizeit, Schule, Hobby, Hausgebrauch (Small Business) und Beruf (bitte nur größere Zielgruppen). Nicht das Rechenergebnis, sondern die Vermittlung des Wissens, wie man eine derartige Aufgabe löst, ist wichtig. Nicht die "Grafik zum anschauen", sondern die Grafik zum Einbauen, zum Abändern auf die eigenen Belange (und wie man so etwas macht) ist gefragt. Schießspiele sind out. Abenteuerspiele, Lernspiele und Denkspiele sind gefragt, sofern sie sich nicht viel besser mit Karten oder per Brett spielen lassen. Nicht vergessen: das Ganze soll auch Spaß machen.
8. Bewertet werden Ihre Programme von unseren Testern u.a. nach folgenden Kriterien: a) Originalität der Programmieridee. b) Optik der grafischen Darbietung, c) Nutzwert des Programmes im Verhältnis zu Programmumfang und Ablaufgeschwindigkeit. d) Nutzung der akustisch sinnvollen Möglichkeiten. e) Decken sich die in der Dokumentation aufgeführten Möglichkeiten mit denen des Programms? f) Subroutinen und Utilitys sollten sich mit einem beigefügten Testprogramm probeweise starten lassen.
9. Honoriert werden nicht nur Listing und Beschreibung, sondern auch eingereichte Hardcopys, Zeichnungen und Bildschirmfotos, sofern sie veröffentlicht werden. Honoriert werden für die Veröffentlichung in CHIP-Special pro Druckseite zwischen 100 und 150 Mark. In diesem Preis ist der Ankauf der Kassette bzw. Diskette sowie das Anbieten des Leserservices (Heftinhalt auf Datenträger zu beziehen) enthalten. Abgezogen am Honorar wird jegliches "Zeilenschinden".
10. Das Urheberrecht für dieses Programm bleibt bei Ihnen. Wir kaufen das ausschließliche Nutzungsrecht für Vervielfältigung und Vertrieb in Europa. Wir nehmen nur Programme an, die noch nirgendwo veröffentlicht wurden.
11. Nachträgliche Änderungen an Ihrem Programm, auch wenn sie erst nach Veröffentlichung aufgrund von Leseranfragen zustande kommen, sollten Sie uns stehts umgehend mitteilen. Auch das gehört zum Service am Leser.

Buch-Tips

HX-20 Dokumentation
Kein Lehrbuch. Diese Dokumentation zeigt die Möglichkeiten des HX-20 auf. Die vorliegende 9. Auflage enthält u.a. detaillierte Beschreibungen der Betriebssystem-Routinen, die für den Programmierer interessant sind. Autor: Lutz Ahrens. 9. Auflage 1984. 93 DIN-A-4-Seiten. Herausgeber: Computerclub EPSILON, Postfach 185, CH-8704 Herrliberg. 100 sfr.

HX-20 Assembler
Einführung in die Assembler-Sprache des HX-20 mit beigepackter Demo-Kassette. Herausgeber: Symic Gesellschaft für Systemberatung Mikrocomputer GmbH, Möchengladbach. 1. Auflage 1984. 216 Seiten DIN-A-5-Format. 248 Mark.

HX-20 BASIC Handbuch
Mit mehr als 150 Programmen. 1. Auflage 1984. 240 Seiten. Herausgeber: Elektra Verlag, vormals Karamanolis-Verlag, Neubiberg b. München. 39 Mark.

Programmieren in EBASIC am Beispiel HX-20
Inklusive Hardware-Beschreibung und Programmierlehrgang mit ausführlichen Beispielen aus der Praxis. Autoren: Wolfgang Grams, Gerd-D. Müller, Markus Nowoczyn. Herausgeber: Datec Computersysteme GmbH, Düsseldorf. 1. Auflage 1982. 680 Seiten DIN-A 4. 79 Mark.

Hardware-Tip

Datenaustausch zwischen HX-20 und CBM-Rechnern
Der HX-20 ist für die mobile Datenerfassung geeignet, da er über die erforderlichen Systemeinheiten verfügt. Mit der eingebauten Schnittstellen kann grundsätzlich ein Daten- und Programmaustausch mit anderen Rechnern erfolgen. Die Anschlußbaugruppe I/O-TOOL HX-20 ist speziell abgestimmt und geeignet zur Verbindung des HX-20 mit einem CBM-Rechner der Serien 30xx, 40xx und 80xx (auch 8032 SK). Hersteller: Görlitz Computerbau, Koblenz. Preis: etwa 600 Mark.

CHIP SPECIAL

1. Auflage 1985

Best.-Nr. 930

Redaktionsdirektor: Richard Kerler

Chefredaktion: Armin Schwarz (verantwortlich für den Inhalt)

Gestaltung: Hans Kuh, Gabi Klotz

Titelillustration: Ezio Geneletti

Mitarbeiter dieser Ausgabe: Franz Joseph Schreiner, Werner Mühleisen

Redaktion: Vogel-Verlag KG Würzburg, Redaktion CHIP-Special, Schillerstr. 23a, D-8000 München 2, Telefon (089) 514930, Telekopierer 535000, Telex 5216449

Verlag: Vogel-Verlag KG, Postfach 6740, D-8700 Würzburg 1, Tel. (0931) 4102-1, Telex 68883, Telefax (0931) 4102-529. Telegramme: CHIP-Würzburg

Verlagsdirektor: Dipl.-Kfm. Herbert Frese

Anzeigenleiter: Harald Kempf, Würzburg (verantwortlich für Anzeigen)

Anzeigenservice: CHIP, Postfach 6740, 8700 Würzburg 1, Tel. (0931) 4102-1, Telex 68883, Michael Belgrad, Durchwahl 4102-433.

PLZ 1–5 und Ausland: Christine Himmer und Wolfgang Hartmann, Durchwahl 4102-227
PLZ 6–8: Angelika Hirsch und Axel Winheim, Durchwahl 4102-513

Vertriebsleitung: Axel Herbschleb, Würzburg

Vertrieb Handelsauflage: Vereinigte Motor-Verlage GmbH & Co. KG, Leuschnerstr. 1, D-7000 Stuttgart 1, Tel. (0711) 2043-1

Bezugsmöglichkeiten: Bestellungen nehmen der Verlag und alle Buchhandlungen im In- und Ausland entgegen. Sollte die Zeitschrift aus Gründen, die nicht vom Verlag zu vertreten sind, nicht geliefert werden können, besteht kein Anspruch auf Nachlieferung oder Erstattung vorausbezahlter Bezugsgelder.

Bankverbindungen Vogel-Verlag:
Dresdner Bank AG, Würzburg
(BLZ 79080052) 3148890000,
Bay. Vereinsbank AG, Würzburg
(BLZ 79020076) 2506173,
Kreissparkasse Würzburg
(BLZ 79050130) 17400,
Postscheckkonto Nürnberg
(BLZ 76010085) 9991-853
Ausland: Postscheckkonto Zürich
8047064,
Niederlande 2662395
Banque Veuve Morin-Pons, Paris
155410314

Gesamtherstellung und Versand: VOGEL-DRUCK WÜRZBURG, Max-Planck-Str. 7/9, D-8700 Würzburg

Unverlangte Manuskripte werden nur zugesandt, wenn Rückporto beigefügt ist.
Für die mit Namen oder Signatur des Verfassers gekennzeichneten Beiträge übernimmt die Redaktion lediglich die presserechtliche Verantwortung.
Die in dieser Zeitschrift veröffentlichten Beiträge sind urheberrechtlich geschützt. Übersetzung, Nachdruck, Vervielfältigung sowie Speicherung in Datenverarbeitungsanlagen nur mit ausdrücklicher Genehmigung des Verlages. Jede im Bereich eines gewerblichen Unternehmens hergestellte oder benutzte Kopie dient gewerblichen Zwecken gem. § 54 (2) UrhG und verpflichtet zur Gebührenzahlung an die VG Wort, Abteilung Wissenschaft, Goethestraße 49, 8000 München 2, von der die Zahlungsmodalitäten zu erfragen sind.
Die Redaktion hat die Manuskripte und Programme sorgfältig geprüft. Für Fehler im Text, in Schaltbildern, Aufbauskizzen, Listings usw. sowie deren Folgen kann keine Haftung übernommen werden. Sämtliche Veröffentlichungen erfolgen ohne Berücksichtigung eines eventuellen Patentschutzes, auch werden Warennamen ohne Gewährleistung einer freien Verwendung benutzt.

Vervollständigen Sie noch heute Ihre Software-Bibliothek!

Bestellen Sie die Ihnen noch fehlenden CHIP-SPECIALS! Mit der nebenstehenden Karte geht's ganz leicht. Einfach ausfüllen und ab die Post!

1 Bestellkarte für alle CHIP-SPECIAL-Programmhefte

☐ **Ja,** senden Sie mir bitte die angekreuzten Specials zu den genannten Preisen zuzüglich Versandkostenanteil DM 3,50 im Inland. (Versandkostenanteil für das Ausland DM 6,-).

Ich bezahle erst, wenn ich Ihre Rechnung erhalten habe.

Datum

Unterschrift

Bitte genaue Anschrift auf der Rückseite angeben.

Anzahl	Titel	Best.-Nr.	DM/St.
	Schneider CPC 464	931	18,–
	Computer-Katalog 1985	932	24,–
	TA alphatronic PC Programme	925	28,–
	PEEK POKE C 64	929	18,–
	C 64 Programme 2	910	18,–
	C 64 Programme 3	919	18,–
	Atari 600 XL/800 XL	920	18,–
	ZX 81 Programme 2	921	18,–
	ZX Spectrum 1	911	18,–
	ZX Spectrum 2	922	18,–
	IBM PC, Kompatible u. IBM PCjr.	917	28,–
	VC 20 Programme 2	909	18,–
	TI-99/4A Programme	906	18,–
	VC 20 Programme 1	754	18,–
	Computer im Selbstbau	901	25,–
	Computerhobby für C 64	924	24,–
	C 64 Unterhaltung + Nutzen	935	14,–

HX-20

2 Wünsche an die Redaktion

HX-20

CHIP SOFT

Bestellen Sie noch heute die Programme für den Epson HX-20

3 Bestellkarte für CHIP SOFT-Epson-HX 20-Programme

Jetzt alle Epson-HX 20-Programme auf Kassette!

HX-20

☐ **Epson HX 20-Programme**

Bitte senden Sie mir alle Programme auf Kassetten für 150,– DM. Schriftverkehr, Rechnungswesen, Mathematik, Bauwesen, Grafik, Tips und Tricks, Haushaltsplan, Finanzen, Lernen, Spiele, Schach, Musik. Jedes dieser 4 Programme kostet 49,95 DM. Alles Inlandspreise inkl. MwSt. zuzügl. 3,50 DM Versand und 3,20 DM Nachnahme.

Name, Vorname

Straße, Postfach

PLZ/Ort

Datum/Unterschrift

Name, Vorname

Straße/Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Antwort

CHIP SPECIAL
Leser-Service 735
Postfach 6740

D-8700 Würzburg 1

CHIP SPECIAL

Name, Vorname

`traße, Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Ich könnte Ihnen für folgende(n) Computer

folgende selbstentworfene Programme anbieten:

Antwort

CHIP SPECIAL
Redaktions-Service
Vogel-Verlag
Schillerstraße 23a

D-8000 München 2

Name, Vorname

Straße, Postfach

PLZ/Ort

Bitte mit 60 Pfennig freimachen

Antwort

CHIP SOFT
Redaktions-Service
Vogel-Verlag
Schillerstraße 23a

D-8000 München 2

CHIP SOFT

In dieser Ausgabe

Schriftverkehr	**Briefkopf** **Text-Star** **Address**
Rechnungswesen	**EDV-Überweisungsvordrucke**
Finanzen	**Ladenkasse** **Schuldtilgung**
Mathematik	**Trigonometrische Funktionen** **Volumen berechnen** **Flächen berechnen**
Bauwesen	**Prismen**
Grafik	**Digitaluhr** **Analoguhr** **Weihnachtsgruß**
Tips und Tricks	**Zeichengenerierung** **HX-Basic** **File-Finder** **While/Wend-Befehle**
Haushaltsplan	**Addition mit Text**
Lernen – Wissen – Können	**Grundrechen-Teacher**
Spiel – Sport – Spaß	**Mein Biorhythmus** **Game of Life** **Code-Gal**
Glücksspiel	**Lottozahlen ermitteln** **PEC-Woman**
Party-Nonsen	**Methusalem**
Schach	**Springer**
Musik	**Loeillet**
Programmierhilfen	**RAM- und ROM-Adressen**